Bayesian filtering for state estimation of the environment

Course: Autonomous Machine Learning
Arman Ahmadian
Seoul National University
Nov. 2016

Contents

Section 1

- Nonlinear transformation
  - EKF, a reliable solution?
  - Nonlinear transformation effects
  - Polar to Cartesian Coordinate Transformation
  - Solution

EKF, a reliable solution?

- Linearized transformations' limitations
  - Only reliable if the error propagation can be **well approximated** by a linear function;
    - At best, **undermines** the performance
    - At worst, causes its estimates to **diverge**.
  - Feasible only if the **Jacobian matrix** exists;
    - **Jump-linear** process models
    - Highly **quantized** sensor measurements
  - Calculating Jacobian matrices can be a very **difficult** and **error-prone** process.

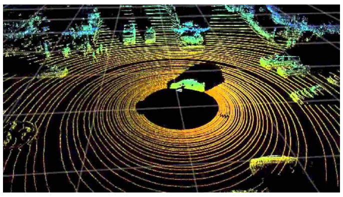
Nonlinear transformation effects

- A random variable $\boldsymbol{x}$
  - mean $\boldsymbol{\mu_x}$
  - covariance $\boldsymbol{\Sigma_x}$.
- A second random variable, $\boldsymbol{z}$

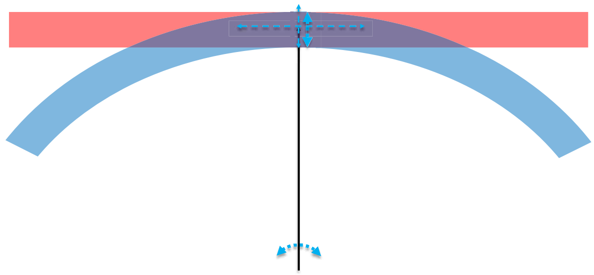
$$\boldsymbol{z} = \boldsymbol{h}[\boldsymbol{x}]$$

- How is the statistics of $\boldsymbol{z}$ characterized?
  - mean $\boldsymbol{\mu_z}$
  - covariance $\boldsymbol{\Sigma_z}$

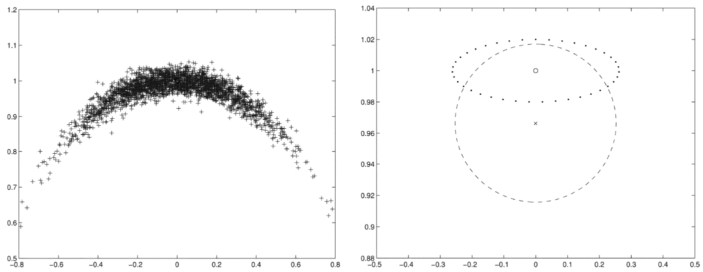
Google's self-driving car



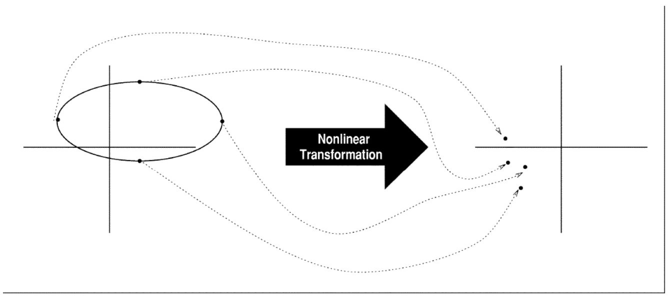
Polar to Cartesian Coordinate Transformations (1)



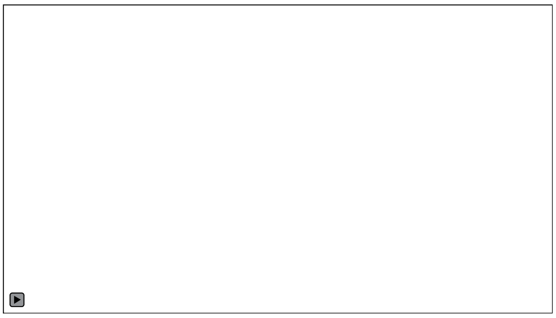
Polar to Cartesian Coordinate Transformations (2)



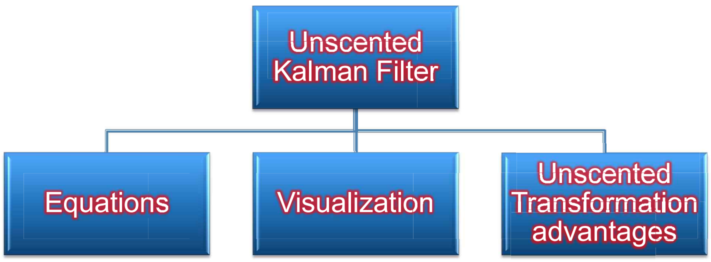
Solution



Google's self-driving car in action



Section 2



Equations

- $2M+1$ points and weights (**sigma point-set**: $\{\chi, w\}$)
- **Scaling parameter**

$$\kappa = 3 - M$$

- Moment-Matching conditions

$$\sum_{i=0}^{2M} w_i \chi_i = \boldsymbol{\mu}$$

$$\sum_{i=0}^{2M} w_i (\chi_i - \boldsymbol{\mu})(\chi_i - \boldsymbol{\mu})^T = \boldsymbol{\Sigma}$$

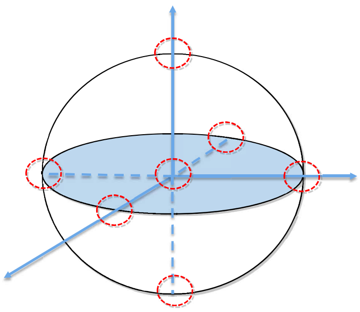
- Solution

$$\chi_0 = \boldsymbol{\mu}, \qquad w_0 = \frac{\kappa}{M+\kappa}$$

$$\chi_{M+i} = \boldsymbol{\mu} + [(M+\kappa)\boldsymbol{\Sigma}]_i^{1/2}, \qquad w_i = \frac{1}{2(M+\kappa)}$$

$$\chi_{M+i} = \boldsymbol{\mu} - [(M+\kappa)\boldsymbol{\Sigma}]_i^{1/2}, \qquad w_i = \frac{1}{2(M+\kappa)}$$

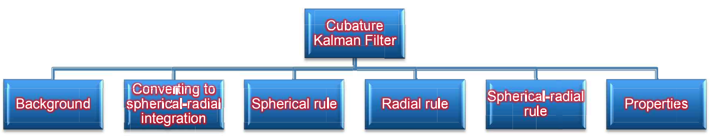
Visualization



Unscented Transformation advantages

- **Black box** filtering library
- The same **computational cost** as the EKF
- Not **sensitive** to the choice of sigma points
- Can be used with **discontinuous transformations**
- pth-order nonlinearity captured ⇨ the first pth-order moments captured

Section 3



Background

- Under the Gaussian assumption, we wish to compute **multidimensional integrals** of a special form

$$\int (\text{nonlinear function}) \times (\text{Gaussian function}) dx$$

Or

$$h(\boldsymbol{f}) = \int_{R_M} f(\boldsymbol{x}) \exp(-\boldsymbol{x}^T \boldsymbol{x})\, d\boldsymbol{x}$$

- How can we efficiently approximate this integral?

Converting to spherical-radial integration

- Change of variables from the **Cartesian** coordinates to **spherical–radial** ones

$$h(\boldsymbol{f}) = \int_0^\infty \int_{\mathcal{U}_M} \boldsymbol{f}(r\boldsymbol{z}) r^{M-1} \exp(-r^2)\, d\boldsymbol{\sigma}\, dr$$

- $\mathcal{U}_M$ is the region defined by $\mathcal{U}_M = \{\boldsymbol{z}; \boldsymbol{z}^T\boldsymbol{z} = 1\}$
- $\sigma(.)$ is a spherical surface measure on $\mathcal{U}_M$

$$\boxed{S(r) = \int_{\mathcal{U}_M} \boldsymbol{f}(r\boldsymbol{z}) d\sigma(\boldsymbol{z})}$$ Spherical rule!

- The integral is converted to

$$\boxed{h(\boldsymbol{f}) = \int_0^\infty S(r) r^{M-1} \exp(-r^2)\, dr}$$ Gaussian quadrature!

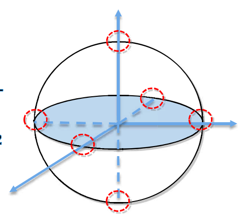
Spherical Rule

- Third-degree spherical rule taking the form

$$\boxed{\int_{\mathcal{U}_M} \boldsymbol{f}(\boldsymbol{z}) d\sigma(\boldsymbol{z}) \approx w \sum_{i=1}^{2M} \boldsymbol{f}[u]_i}$$

- The **cubature points**: at the intersection of an M-dimensional hypersphere and its axes.
- Solving for monomials $\boldsymbol{f}(\boldsymbol{z}) = 1$ and $\boldsymbol{f}(\boldsymbol{z}) = z_1^2$ yields

$$w = \frac{A_M}{2M}$$
$$u^2 = 1$$



Radial Rule

- Gaussian quadrature (most efficient numerical method to compute an integral in a single dimension)

$$\int_D f(x)w(x)dx \approx \sum_{i=1}^{2M} w_i f(x_i)$$

- Where
  - Quadrature points: x_i
  - Quadrature weights: w_i

$$\int_0^\infty f(x)x^{M-1}\exp(-x^2)\,dx = \frac{1}{2}\int_0^\infty f(\sqrt{t})\,t^{\frac{M}{2}-1}\exp(-t)\,dt$$

- first-degree generalized **Gauss–Laguerre rule**!

$$\boxed{\int_0^\infty f_i(x)x^{M-1}\exp(-x^2)\,dt \approx w_1 f_i(x_1)}$$

$$w_1 = \frac{1}{2}\Gamma\left(\frac{M}{2}\right), \qquad x_1 = \sqrt{\frac{M}{2}}$$

Spherical-radial rule (1)

Proposition 1: Let the radial integral be computed numerically by an $\boldsymbol{m_r}$**-point** Gaussian quadrature rule

$$\int_0^\infty f(r)r^{M-1}\exp(-r^2)\,dr = \sum_{i=1}^{m_r} a_i f(r_i)$$

Let the spherical integral be computed numerically by an $\boldsymbol{m_s}$**-point** spherical rule

$$\int_{\mathcal{U}_M} \boldsymbol{f}(r\boldsymbol{s})d\sigma(\boldsymbol{s}) = \sum_{j=1}^{m_s} b_j \boldsymbol{f}(r\boldsymbol{s}_j)$$

Then, an $(m_s \times m_r)$-point spherical–radial cubature rule is approximately given by the double summation

$$\int_{\mathrm{R}_M} \boldsymbol{f}(r)r^{M-1}\exp(-\boldsymbol{x}^T\boldsymbol{x})\,d\boldsymbol{x} = \sum_{j=1}^{m_s}\sum_{i=1}^{m_r} a_i b_j \boldsymbol{f}(r_i\boldsymbol{s}_j)$$

Spherical-radial rule (2)

- We can now numerically compute the **standard Gaussian weighted integral**

$$\boxed{h_N(\boldsymbol{f}) \approx \sum_{i=1}^{2M} w_i \boldsymbol{f}(\xi_i)}$$

where the **cubature-point set** is

$$\left\{\xi_i = \boldsymbol{e}_i\sqrt{M}, w_i = \frac{1}{2M}\right\}_{i=1}^{2M}$$

where $\boldsymbol{e}_i$ is the canonical unit vector.

Properties

- Derivative-free ⇨ **Noise-smoothing** capability
- **Regularization**
- **Inherits well-known properties** of the linear Kalman Filter
  - Square-root filtering
- Complexity
  - **Linear** in the number of function evaluation
  - Complexity grows as M^3
  - Eases the **curse of dimensionality**

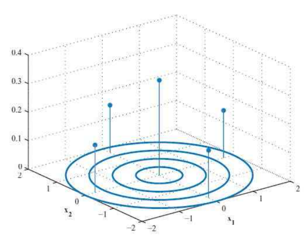
Section 4

Unscented vs. Cubature

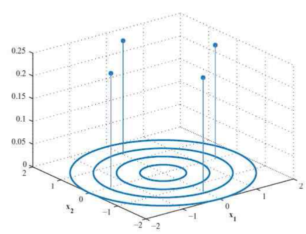
- UKF vs. CKF (1)
- UKF vs. CKF (2)
- The curse of dimensionality

UKF vs. CKF (1)

sigma point-set for the UKF



cubature point-set for the CKF



UKF vs. CKF (2)

UKF

- Heuristic
  - Scaling parameter κ
  - $\kappa = 0$ corresponds to the cubature point-set
- $(2M + 1)$ sigma points
  - One point at the origin and the rest symmetrically distributed on the surface of a 2M-dimensional ellipsoid
  - The presence of a weighted sigma-point at the origin weakens its approximating power
- Curse of dimensionality problem
  - Odd set of weighted sigma-points

CKF

- Mathematically rigorous
  - No scaling parameter
- $2M$ cubature points
  - Symmetrically distributed on the surface of a 2M-dimensional ellipsoid
- Curse of dimensionality problem
  - Even set of weighted cubature points

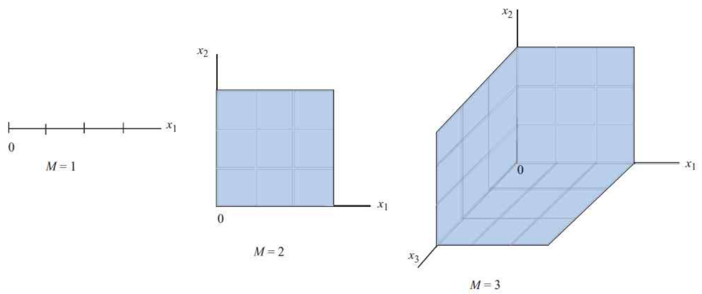
The curse of dimensionality

- Computational complexity of the state-estimation problem grows **exponentially** with increasing **dimensionality** of the state-space model.
- Stochastic differential equation

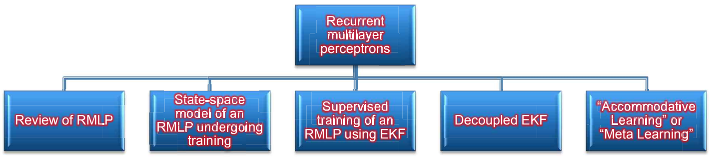
$$\frac{d}{dt}\boldsymbol{x}(t) = \boldsymbol{a}\big(\boldsymbol{x}(t-1)\big) + \boldsymbol{\omega}(t)$$

- When **discretizing** the problem, we should assign a certain number of **quantization levels** to each dimension!

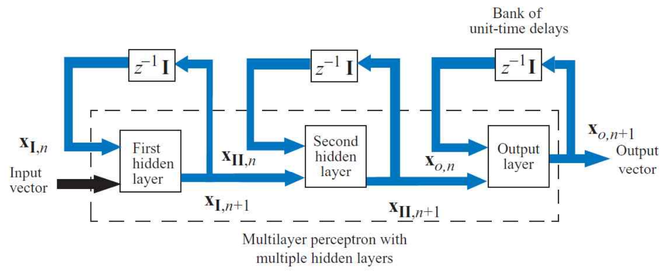
The curse of dimensionality



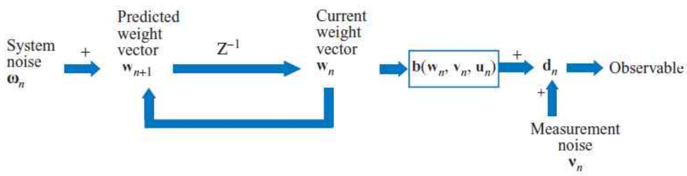
Section 5



Review of RMLP



State-space model of an RMLP undergoing training (2)



State-space model of an RMLP undergoing training (1)

- Let the vector $\boldsymbol{w}_n$ denote the synaptic weights
- State-space model of the network
  - System equation

$$\boldsymbol{w}_n = \boldsymbol{w}_n + \boldsymbol{\omega}_n$$

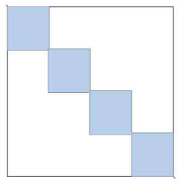
  - Measurement equation

$$\boldsymbol{d}_n = \boldsymbol{b}(\boldsymbol{w}_n, \boldsymbol{v}_n, \boldsymbol{u}_n) + \boldsymbol{\nu}_n$$

- where
  - The **dynamic noise** $\boldsymbol{\omega}_n$ is of covariance matrix $\boldsymbol{Q}_\omega$.
    - To **anneal** the supervised training
  - The **measurement noise** $\boldsymbol{\nu}_n$ is of diagonal covariance matrix $\boldsymbol{R}_n$.
  - $\boldsymbol{d}_n$ is the observable.
  - $\boldsymbol{u}_n$ denotes the input signal.
  - $\boldsymbol{v}_n$ represents the recurrent node activities inside the network.

Decoupled EKF

- How to reduce the complexity of EKF?
- Ignore the interactions between the estimates of certain weights in the recurrent neural network by partitioning them into g disjoint weight groups
- Mutually exclusive weight groups ⇨ block diagonal covariance matrix
- p output nodes and W weights
- Complexity of EKF
  - Computational complexity: $O(pW^2)$
  - Storage requirements: $O(W^2)$
- Complexity of DEKF
  - Computational complexity: $O\left(pW^2 + p\sum_{i=1}^{g} W_i^2\right)$
  - Storage requirements: $O\left(\sum_{i=1}^{g} W_i^2\right)$

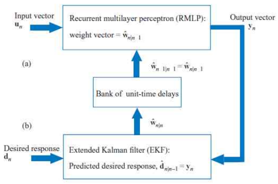


Supervised training of an RMLP using EKF (1)

- Training sample: $\{\boldsymbol{u}_n, \boldsymbol{d}_n\}_{n=1}^{N}$
- The weight update

$$\widehat{\boldsymbol{w}}_{n|n} = \widehat{\boldsymbol{w}}_{n|n-1} + \boldsymbol{G}_n(\boldsymbol{d}_n - \boldsymbol{b}(\widehat{\boldsymbol{w}}_{n-1}, \boldsymbol{v}_n, \boldsymbol{u}_n))$$

- nonlinear RMLP ⇨ nonlinear sequential state-estimator.

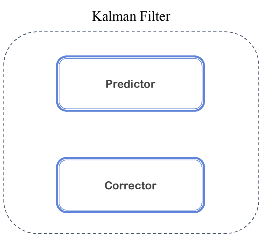


"Accommodative Learning" or "Meta Learning"

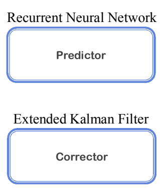
- Consider a recurrent neural network embedded in a stochastic environment with relatively small variability in its statistical behavior. Provided that the underlying **probability distribution** of the environment is fully represented in the supervised-training sample supplied to the network, it is possible for the network to adapt to the relatively small statistical variations in the environment **without** any further **on-line adjustments** being made to the synaptic weights of the network.
- Only valid for **recurrent networks** because dynamic state of a recurrent network actually acts as a "short-term memory" that carries an estimate or statistic of the uncertain environment for adaptation in which the network is embodied.

Supervised training of an RMLP using EKF (3)

Traditional applications of the Kalman filter



Supervised training of a recurrent NN



Conclusion

- **Linearization** in the EKF causes the estimate to be biased and inaccurate;
- UKF and CKF improve the accuracy of nonlinear estimation using **a set of points**
  - Derivation of CKF is **mathematically rigorous** while that of **UKF is heuristic**;
  - UKF uses an **odd** set of points whereas CKF uses an **even** set of points;
  - UKF has a **scaling parameter** that needs to be set whereas CKF **does not**;
- **CKF is better** than UKF in terms of accuracy and complexity!
- **Recurrent Neural Networks** can be trained using a nonlinear state estimator
- RMLPs can **adapt to small changes** in the environment provided that the statistics of the environment has been given to the network while training.

The End

Thank you!
Any questions?