Implementation of Adaptive Kernel Kalman Filter in Stone Soup

James S. Wright, James R. Hopgood, Mike E. Davies, Ian K. Proudler, Mengwei Sun

Introduction

- Adaptive Kernel Kalman Filter
- This is the first open source implementation of the AKKF
Introduction: Outline

- Stone Soup: Tracking and State Estimation Framework
- Adaptive Kernel Kalman Filter
- Components
- Examples
- Conclusions
What is Stone Soup?

Framework (Open Source)
Stone Soup: Components

- Readers
- Simulators
- Feeders
- Algorithmic Components
  - Predictors
  - Updaters
- Metrics
- Writers
Stone Soup: Algorithmic Components

Diagram showing the components of the algorithmic model:

- Transition Model
- Measurement Model
- Ground Truth Simulator
- Detection Simulator

Components:
- Tracker
- Associator
- Hypothesiser
- Initiator
- Predictor
- Updater
- Deleter

Dependencies:
- stonesoup.updater.base.Updater
- stonesoup.updater.kalman.KalmanUpdater
- stonesoup.updater.kalman.ExtendedKalmanUpdater
- stonesoup.updater.kalman.UnscentedKalmanUpdater
- stonesoup.updater.particle.ParticleUpdater
Adaptive Kernel Kalman Filter (AKKF)
Components

- **States**
  - KernelParticleState

- **Kernels**
  - Kernel, QuadraticKernel, QuarticKernel, GaussianKernel

- **Predictors**
  - AdaptiveKernelKalmanPredictor

- **Updaters**
  - AdaptiveKernelKalmanUpdater
KernelParticleState

Inherits the functionality of ParticleState and adds the kernel covariance defined by the following equations

\[
S_k^- = \Gamma_k S_{k-1}^+ \Gamma_k^T + V_k
\]

\[
S_k^+ = S_k^- - Q_k G_{yy} S_k^-
\]
### Components: Kernels

- **Kernel**
  \[ k(x, x') = x^T x' \]

- **QuadraticKernel**
  \[ k(x, x') = (\alpha \langle x, x' \rangle + c)^2 \]

- **QuarticKernel**
  \[ k(x, x') = (\alpha \langle x, x' \rangle + c)^4 \]

- **GaussianKernel**
  \[ k(x, x') = \exp \left( \frac{||x - x'||^2}{2\sigma^2} \right) \]
AdaptiveKernelKalmanPredictor

Inherits the functionality of KalmanPredictor and adds the functionality of translating between state space and kernel space.

AdaptiveKernelKalmanUpdater

Inherits the functionality of KalmanUpdater and adds the functionality of translating between measurement space and kernel space.
Example 1. Applying the Quadratic Kernel.
Example 2: Applying the Quadratic Kernel
Example 3: Which Kernel?
Conclusions

- AKKF outperforms PF for a given number of particles
- We provided an open source implementation of the AKKF in Stone Soup
- Via inheritance, modularity, and encapsulation, we have demonstrated that Stone Soup is an efficient way of enacting, testing and demonstrating the efficacy of novel tracking algorithms
- We have shown a method of integrating newer algorithms into Stone Soup and a workflow to support other algorithms being implemented in Stone Soup

Get in touch:
https://github.com/dstl/Stone-Soup  StoneSoup@dstl.gov.uk