Package: RcppKalman (via r-universe)

August 11, 2024

Type Package

Title 'RcppArmadillo'-Based Kalman Filtering

Version 0.0.6.1

Date 2018-12-23

Author Dirk Eddelbuettel

Maintainer Dirk Eddelbuettel <edd@debian.org>

Description An 'RcppArmadillo'-based port of the Kalman filtering code in the 'EKF/UKF Toolbox for Matlab' by Simo Särkkä, Jouni Hartikainen, and Arno Solin is provided.
Note that this package is at this point still incomplete, but contains two demo functions replicating demos in 'EKF/UKF'.

License GPL (>= 2)

Imports expm, Rcpp

LinkingTo Rcpp, RcppArmadillo

Suggests xts

Encoding UTF-8

RoxygenNote 5.0.1

Repository https://eddelbuettel.r-universe.dev

RemoteUrl https://github.com/eddelbuettel/rcppkalman

RemoteRef HEAD

RemoteSha 129f01c00892b22b50893dc94deb43573be10723

Contents

`RcppKalman-package` *RcppArmadillo based Kalman filteringKalman Filtering in C++ accessible to R via Rcpp*

Description

This package provides R with a C++ port of some functions from the EKF/UKF Toolbox for Matlab by Simo Särkkä, Jouni Hartikainen, and Arno Solin.

Details

TBD

Author(s)

The EKF/UKF Toolbox for Matlab was written by Simo Särkkä, Jouni Hartikainen, and Arno Solin. Dirk Eddelbuettel wrote this package, and maintains it.

References

See http://becs.aalto.fi/en/research/bayes/ekfukf/ for the EKF/UKF Toolbox.

`expm` *Compute the exponential of a matrix*

Description

This function computes the exponential of a matrix.

Usage

```
expm(x)
```

Arguments

`x`	An numeric matrix

Details

This functions calls the `expm` function from the eponymous package **expm**. This is implemented via a registered function call, and does not required explicit linking at the C level. However, the **expm** package is imported in order to access its registered function at the C level.

As the documentation of package **expm** states, the underlying implementation borrows from the **Matrix** package which itself takes it from GNU Octave.

Value

A numeric matrix

Author(s)

Dirk Eddelbuettel

See Also

The **expm** package and its documentation.

Examples

```
## example is from the vignette in package expm
M <- matrix(c(4, 1, 1, 2, 4, 1, 0, 1, 4), 3, 3)

## expected output
expM <- matrix(c(147.8666, 127.7811, 127.7811, 183.7651, 183.7651,
                 163.6796, 71.79703, 91.88257, 111.96811), 3, 3)

## we only have the expected result to about six digits
all.equal(expm(M), expM, tolerance=1.0e-6)
```

---

kfPredict *Kalman Filter Prediction step*

Description

This function performs the Kalman Filter prediction step

Usage

```
kfPredict(x, P, A, Q, B, u)
```

Arguments

x	An N x 1 mean state estimate of previous step
P	An N x N state covariance of previous step
A	(Optional, default idendity) transition matrix of the discrete model
Q	(Optional, default zero) process noise of discrete model
B	(Optional, default idendity) input effect matrix
u	(Optional, default empty) constant input

Value

A list with two elements

X the predicted state mean, and

P the predicted state covariance.

Author(s)

The EKF/UKF Toolbox was written by Simo Särkkä, Jouni Hartikainen, and Arno Solin.

Dirk Eddelbuettel is porting this package to R and C++, and maintaing it.

See Also

kfUpdate, ltiDisc and the documentation for the EKF/UKF toolbox at http://becs.aalto.fi/en/research/bayes/ekfukf

---

kfUpdate *Kalman Filter measurement update step*

Description

This function performs the Kalman Filter measurement update step

Usage

```
kfUpdate(x, P, y, H, R)
```

Arguments

x	An N x 1 mean state estimate after prediction step
P	An N x N state covariance after prediction step
y	A D x 1 measurement vector.
H	Measurement matrix.
R	Measurement noise covariance.

Details

This functions performs the Kalman Filter measurement update step.

Value

A list with elements

X the update state mean,

P the update state covariance,

K the computed Kalman gain,

IM the mean of the predictive distribution of Y, and

IS the covariance of the predictive distribution of Y

Author(s)

The EKF/UKF Toolbox was written by Simo Särkkä, Jouni Hartikainen, and Arno Solin.

Dirk Eddelbuettel is porting this package to R and C++, and maintaing it.

See Also

kfPredict and the documentation for the EKF/UKF toolbox at http://becs.aalto.fi/en/research/bayes/ekfukf

---

ltiDisc — *Discretize Linear Time-Invariant ODE*

Description

Discretize Linear Time-Invariant ODE with Gaussian Noise

Usage

```
ltiDisc(F, L, Q, dt)
```

Arguments

F	An N x N feedback matrix
L	(Optional, default idendity) N x L noise effect matrix
Q	(Optionalm default zeros) L x L diagonal spectral density
dt	(Option, default one) time step

Details

This function discretizes the linear time-invariant (LTI) ordinary differential equation (ODE).

Value

A list with elements

A the transition matrix, and

Q the discrete process covariance

Author(s)

The EKF/UKF Toolbox was written by Simo Särkkä, Jouni Hartikainen, and Arno Solin.

Dirk Eddelbuettel is porting this package to R and C++, and maintaing it.

See Also

The documentation for the EKF/UKF toolbox at http://becs.aalto.fi/en/research/bayes/ekfukf

`rtsSmoother` *Rauch-Tung-Striebel smoother*

Description

This function computes the Rauch-Tung-Striebel smoother.

Usage

```
rtsSmoother(M, P, A, Q)
```

Arguments

`M`	An N x K matrix of K mean estimates from the Kalman Filter
`P`	An N x N x K cube length K with N x N state covariances matrices from the Kalman Filter
`A`	An N x N state transition matrix (or in the more general case a list of K such matrices; not yet implemented)
`Q`	An N x N noise covariance matrix (or in the more general case a list of K such matrices; not yet implemented)

Details

This function implements the Rauch-Tung-Striebel smoother algorithm which calculate a "smoothed" sequence from the given Kalman filter output sequence by conditioning all steps to all measurements.

Value

A list with three elements

SM the smoothed mean sequence,

SP the smooted state covariance sequence,and

D the smoothed gain sequence.

Author(s)

The EKF/UKF Toolbox was written by Simo Särkkä, Jouni Hartikainen, and Arno Solin.

Dirk Eddelbuettel is porting this package to R and C++, and maintaing it.

See Also

kfPredict, kfUpdate, and the documentation for the EKF/UKF toolbox at `http://becs.aalto.fi/en/research/bayes/ekfukf`

`tfSmoother` *Two-filter Smoother*

Description

This function computes the 'Two filter-based' Smoother

Usage

```
tfSmoother(M, P, Y, A, Q, H, R, useinf)
```

Arguments

`M`	An N x K matrix of K mean estimates from Kalman filter
`P`	An N x N x K matrix of K state covariances from Kalman Filter
`Y`	A D x K matrix of K measurement sequences
`A`	A N x N state transition matrix.
`Q`	A N x N process noise covariance matrix.
`H`	A D x N measurement matrix.
`R`	A D x D measurement noise covariance.
`useinf`	An optional boolean variable indicating if information filter should be used (with default `true`).

Details

This function implements the two filter linear smoother which calculates a "smoothed" sequence from the given Kalman filter output sequence by conditioning all steps to all measurements.

Value

A list with two elements

M the smoothed state mean sequence, and

P the smoothes state covariance sequence.

Author(s)

The EKF/UKF Toolbox was written by Simo Särkkä, Jouni Hartikainen, and Arno Solin.

Dirk Eddelbuettel is porting this package to R and C++, and maintaing it.

See Also

kfPredict, kfUpdate, and the documentation for the EKF/UKF toolbox at `http://becs.aalto.fi/en/research/bayes/ekfukf`

Index