`robKalman` — A package on Robust Kalman Filtering

Peter Ruckdeschel, Fakultät für Mathematik und Physik, Univ. Bayreuth and

Bernhard Spangl, Universität für Bodenkultur, Wien

Abstract

We want to discuss a proposal on an implementation of Robust Kalman filtering based on `S4` classes. To do so, we are geared to the existing implementations of the Kalman filter from the basic `R` distribution (cf. [5] and [1]) as well as from the bundle `dse` (cf. [2]). By means of the `setOldClass` mechanism (cf. [5]), we register existing `S3` classes from these implementations as `S4` classes and extend them for our purposes. As generic functions we will present implementations of the classical Kalman filter, the ACM filter from [3], and the rLS-filter from [4].

Robust Kalman Filtering

Classical Setup

Linear State-Space-Models (SSMs)

- state equation: $\boxed{X_t = F_t X_{t-1} + v_t}$ observation equation: $\boxed{Y_t = Z_t X_t + \varepsilon_t}$
- Ideal model: $X_0 \sim \mathcal{N}_p(a_0, \Sigma_0)$, $v_t \sim \mathcal{N}_p(0, Q_t)$, $\varepsilon_t \sim \mathcal{N}_q(0, V_t)$, all independent
- (preliminary ?) simplification: Hyper parameters F_t, Z_t, V_t, Q_t constant in t
- Problem: Reconststruction of X_t by means of $Y_s, s \le t$
- Criterium: MSE $\rightsquigarrow$ general solution: $\mathrm{E}\, X_t | (Y_s)_{s \le t}$
- Computational difficulties: $\Longrightarrow$ restriction to **linear** proced.s/ or: Gaussian assumptions

Kalman filter

0. Initialization ($t = 0$): $X_{0|0} = a_0, \quad \Sigma_{0|0} = \Sigma_0$
1. Prediction ($t \ge 1$): $X_{t|t-1} = F X_{t-1|t-1}, \quad \mathrm{Cov}(X_{t|t-1}) = \Sigma_{t|t-1} = F \Sigma_{t-1|t-1} F' + Q$
2. Correction ($t \ge 1$): $X_{t|t} = X_{t|t-1} + K_t (Y_t - Z X_{t|t-1}), \quad$ for $K_t = \Sigma_{t|t-1} Z' (Z \Sigma_{t|t-1} Z')^{-}$ (Kalman gain), $\mathrm{Cov}(X_{t|t}) = \Sigma_{t|t} = \Sigma_{t|t-1} - K_t Z \Sigma_{t|t-1}$

Robustification

Types of outliers and robustification

- IOs (system intrinsic): state equation is distorted — not considered here
- AO/SOs (exogeneous): observations are distorted: either error ε_t is affected (AO) / or observations Y_t are modified (SO)
- a robustifications as to AO/SOs is to
  - retain recursivity (three-step approach)
  - modify correction step $\rightsquigarrow$ bound influence of Y_t

Considered approach I: Approximate conditional mean (ACM): Martin[79]

- $\dim Y_t = 1$, particular model: $Y_t \sim \mathrm{AR}(p) \rightsquigarrow X_t = (Y_t, \ldots, Y_{t-p+1})$, hyper parameters $Z = (1, 0, \ldots, 0)$, $V^{\mathrm{id}} = 0$, F, Q unknown
- estimation of F, Q by means of GM-Estimators
- modified Corr.step: for suitable location influence curve ψ

$$X_{t|t} = X_{t|t-1} + \Sigma_{t|t-1} Z' \psi(Y_t - Z X_{t|t-1}), \quad \Sigma_{t|t} = \Sigma_{t|t-1} - \Sigma_{t|t-1} Z' \psi'(Y_t - Z X_{t|t-1}) Z \Sigma_{t|t-1}$$

Considered approaches II: rLS filter: (P.R.[01])

- $\dim X_t, \dim Y_t$ arbitrary, finite
- assumes hyper parameters a_0, Z, V^{id}, F, Q known
- modified Corr.step: for Euclidean norm $|\cdot|$

$$X_{t|t} = X_{t|t-1} + H_b\big(K_t(Y_t - Z X_{t|t-1})\big), \qquad H_b(X) = X \min\{1, b/|X|\}$$

- optimality for SO's in some sense

Implementation proposal

Concept / Strategy for package `robKalman`

Contents

- Kalman filter: filter, Kalman gain, covariances
- ACM-filter: filter, GM-estimator
- rLS-filter: filter, calibration of clipping height
- further recursive filters?
  $\rightsquigarrow$ general interface `recursiveFilter` with Arguments: state space model (hyper parameters) and functions for the init./pred./corr.step

Concept

- Programming language: completely in `S`
- Use existing infrastructure (time series classes) for: graphics, diagnostics, management of date/time
- Split user interface (`S4`-objects) and "Kalman code" (no `S4`-objects)
- Use of `S4`:
  - Classes for state space models(SSMs), filter results, control structures (tuning parameters)
  - Methods: filters, accessor/replacement fcts, `simulate` for SSMs filter diagnostics

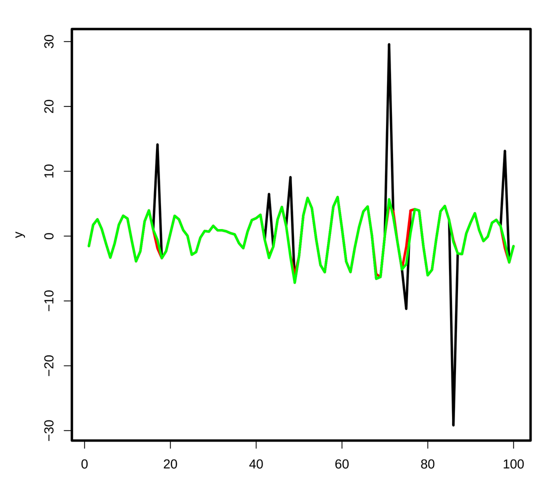
Implementation so far

Demonstration: `ACMfilt`

```
## generation of data from AO model:
Eps ← as.ts(rnorm(100))
ar2 ← arima.sim(list(ar = c(1, −0.9)),
        100, innov = Eps)
Binom ← rbinom(100, 1, 0.1)
Noise ← rnorm(100,sd = 10)
y ← ar2 + as.ts(Binom*Noise)

## determination of GM-estimates
y.arGM ← arGM(y, 3)
## ACM-filter
y.ACMfilt ← ACMfilt(y, y.arGM)

plot(y); lines(y.ACMfilt$filt , col=2)
lines(ar2,col="green")
```



green: ideal time series, black: AO contam. time series, red: result ACM

Demonstration: `rLSfilter`

```
## specification of SSM: (p=2, q=1)
a0  ← c(1, 0); S0  ← matrix(0, 2, 2)
F   ← matrix(c(.7, 0.5, 0.2, 0), 2, 2); Q  ← matrix(c(2, 0.5, 0.5, 1), 2, 2)
Z   ← matrix(c(1, −0.5), 1, 2); Vi  ← 1; TT ← 50 ## (time horizon)

## AO-contamination
mc  ← −20; Vc  ← 0.1; ract ← 0.1

#Simulation::
X  ← simulateState(a, S0, F, Q, TT)
Yid  ← simulateObs(X, Z, Vi, mc, Vc, r=0); Yre  ← simulateObs(X, Z, Vi, mc, Vc, ract)

### calibration b
#limiting S_{t|t-1}

SS ← limitS(S, F, Q, Z, Vi)
# by efficiency in the ideal model / by contamination radius
(B1 ← rLScalibrateB(Z, SS, Vi, eff=.9)); (B2 ← rLScalibrateB(Z, SS, Vi, r=.1))

### evaluation of rLS
f1.id ← rLSFilter(Yid,a,Ss,F,Q,Z,Vi,B1$b); f1.re ← rLSFilter(Yre,a,Ss,F,Q,Z,Vi,B1$b)
f2.id ← rLSFilter(Yid,a,Ss,F,Q,Z,Vi,B2$b); f2.re ← rLSFilter(Yre,a,Ss,F,Q,Z,Vi,B2$b)
## Result: behind references
```

References:

[1]. Durbin, J. and Koopman, S. J.(2001): *Time Series Analysis by State Space Methods.* Oxford University Press.

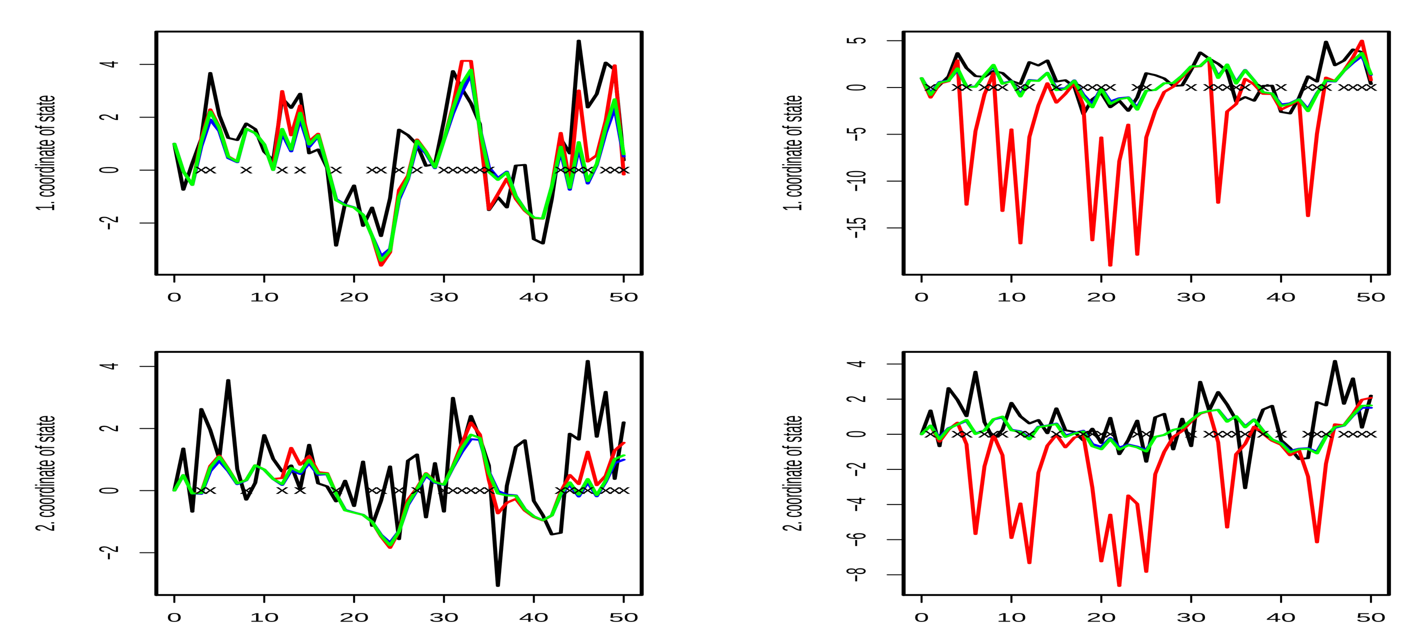
[2]. Gilbert, P. (2005): Brief User's Guide: Dynamic Systems Estimation (DSE). Available in the file `doc/dse-guide.pdf` distributed together with the `R` bundle `dse`, to be downloaded from `http://cran.r-project.org`

[3]. Martin, D. (1979): Approximate conditional-mean type smoothers and interpolators. In *Smoothing techniques for curve estimation, Proc. Workshop, Heidelberg 1979*, Lect. Notes Math. 757, p. 117-143

[4]. Ruckdeschel, P. (2001): Ansätze zur Robustifizierung des Kalman Filters. Bayreuther Mathematische Schriften, Vol. 64.

[5]. R Development Core Team (2005): R:*A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org`

left: ideal situation /right: AO-contaminated situation



black: real state, red: class. Kalman filter green: rLS filter (B1), blue: rLS filter (B2)