# robKalman — a package on Robust Kalman Filtering

Peter Ruckdeschel[1] Bernhard Spangl[2]

1 UNIVERSITÄT BAYREUTH Fakultät für Mathematik und Physik

Peter.Ruckdeschel@uni-bayreuth.de

www.uni-bayreuth.de/departments/math/org/mathe7/RUCKDESCHEL

2 Universität für Bodenkultur, Wien

Bernhard.Spangl@boku.ac.at

www.rali.boku.ac.at/statedv.html

useR! 16.06.2006

# Classical Setup: Linear State-Space-Models

- 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 assumption:

$$X_0 \sim \mathcal{N}_p(a_0, \Sigma_0), \quad v_t \sim \mathcal{N}_p(0, Q_t), \quad \varepsilon_t \sim \mathcal{N}_q(0, V_t),$$

  all independent

- (preliminary ?) simplification: Hyper parameters $F_t, Z_t, V_t, Q_t$ constant in $t$

# Problem and classical solution

- Problem: Reconststruction of $X_t$ by means of $Y_s, s \leq t$
- Criterium: MSE
- $\leadsto$ general solution: $\mathrm{E}\, X_t|(Y_s)_{s\leq t}$
- Computational difficulties:
  $\Longrightarrow$ restriction to **linear** procedures
  / or: Gaussian assumptions
- $\leadsto$ classical **Kalman Filter**

# Kalman filter

0. Initialization ($t = 0$):

$$X_{0|0} = a_0, \quad \Sigma_{0|0} = \Sigma_0$$

1. Prediction ($t \geq 1$):

$$X_{t|t-1} = FX_{t-1|t-1}, \quad \mathrm{Cov}(X_{t|t-1}) = \Sigma_{t|t-1} = F\Sigma_{t-1|t-1}F' + Q$$

2. Correction ($t \geq 1$):

$$\begin{array}{rcl} X_{t|t} & = & X_{t|t-1} + K_t(Y_t - ZX_{t|t-1}) \\ K_t & = & \Sigma_{t|t-1}Z'(Z\Sigma_{t|t-1}Z')^{-}, \qquad \text{(Kalman gain)} \\ \mathrm{Cov}(X_{t|t}) & = & \Sigma_{t|t} = \Sigma_{t|t-1} - K_t Z \Sigma_{t|t-1} \end{array}$$

# Types of outliers and robustification

- IOs (system intrinsic): state equation is distorted
  — not considered here
- AO/SOs (exogeneous): observations are distorted:
  - either error $\varepsilon_t$ 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 $Y_t$
  - retain init./pred.step but with modified filter past $X_{t-1|t-1}$

# Considered approaches

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 $\mathrm{GM}$-Estimators
- modified Corr.step: for suitable location influence curve $\psi$

$$
\begin{aligned}
X_{t|t} &= X_{t|t-1} + \Sigma_{t|t-1} Z' \psi(Y_t - Z X_{t|t-1}) \\
\Sigma_{t|t} &= \Sigma_{t|t-1} - \Sigma_{t|t-1} Z' \psi'(Y_t - Z X_{t|t-1}) Z \Sigma_{t|t-1}
\end{aligned}
$$

# Considered approaches II

rLS filter: [P.R.(01)]

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

$$
\begin{aligned}
X_{t|t} &= X_{t|t-1} + H_b\big(K_t(Y_t - ZX_{t|t-1})\big) \\
H_b(X) &= X \min\{1, b/|X|\} \qquad \text{for } |\cdot| \text{ Euclidean norm}
\end{aligned}
$$

- optimality for SO's in some sense

# Concept and strategy

Goal: 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)
  - functions for the init./pred./corr.step

# Concept and strategy II

- Programming language
  - completely in `S`
  - perhaps some code in `C` (much) later
- Use existing infrastructure
  - package candidates
    - One dimensional: `KalmanLike` (package `stats`); time series classes: `ts`, `its`, `irts`, `zoo`, `zoo.reg`
    - Multivariate setting: `dse` bundle by Paul Gilbert; perhaps `zoo`?
  - use for: graphics, diagnostics, management of date/time
- Split user interface and "Kalman code"
  - internal functions: no `S4`-objects
  - user interface: `S4`-objects

# Concept and strategy III

- Use of `S4`
  - Hierarchic Classes:
    - state space models (SSMs) (Hyper-Parameter, distributional assumptions, outlier types)
    - filter results (specific subclass of (multivariate) time series)
    - control structures for filters (tuning parameters)
  - Methods:
    - filters (for different types of SSMs)
    - accessor/replacement functions
    - `simulate` for SSMs
    - filter diagnostics: `getClippings`, `conf.intervals` ?
    - tests?
  - constructors/generating funtions

# Implementation so far: interfaces

- preliminary, "`S4`-free" interfaces
  - Kalman filter (in our context) `KalmanFilter`
  - rLS (P.R.): `rLSFilter`
    — with routines for calibration at given
    - efficency in ideal model
    - contamination radius
  - ACM (B.S.) `ACMfilt`, `ACMfilter`
    - with function `arGM` for AR-parameters by GM-estimates
    - various $\psi$-functions are available:
      Hampel (ACM-filter), Huber, Tukey (both GM-estimators)
      —see `?.psi`
  - all: wrappers to `recursiveFilter`

# Implementation so far: package `robKalman`

- package `robKalman`
  - routines gathered in package `robKalman`, version `0.1`
  - documentation
  - demos
- required packages — all available from `CRAN`: `methods`, `graphics`, `startupmsg`, `dse1`, `dse2`, `MASS`, `limma`, `robustbase`
- availability: web-page setup under

```
http://www.uni-bayreuth.de/departments/
       /math/org/mathe7/robKalman/
```

# Next steps

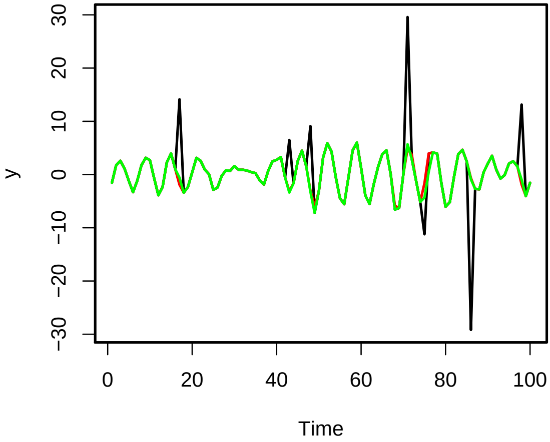
- OOP
  - definition of `S4` classes
    $\rightsquigarrow$ close contact to
    - `RCore`,
    - Paul Gilbert,
    - possibly Gabor Grothendiek and Achim Zeileis (`zoo`)
  - casting/conversion functions for various time series classes
- User interface `robfilter` (?)
  - goal: four arguments: data, SSM, control-structure, filter type
  - should take various definitions of SSMs, data in various time series classes,
  - possibly simpler interfaces for ACM $\rightsquigarrow$ `ACMfilt`-like
- Release Schedule
  - wait for results of discussion as to class definition
  - guess: end of 2006

# Demonstration: ACMfilt

```
## generation of data from AO model:
set.seed(361)
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

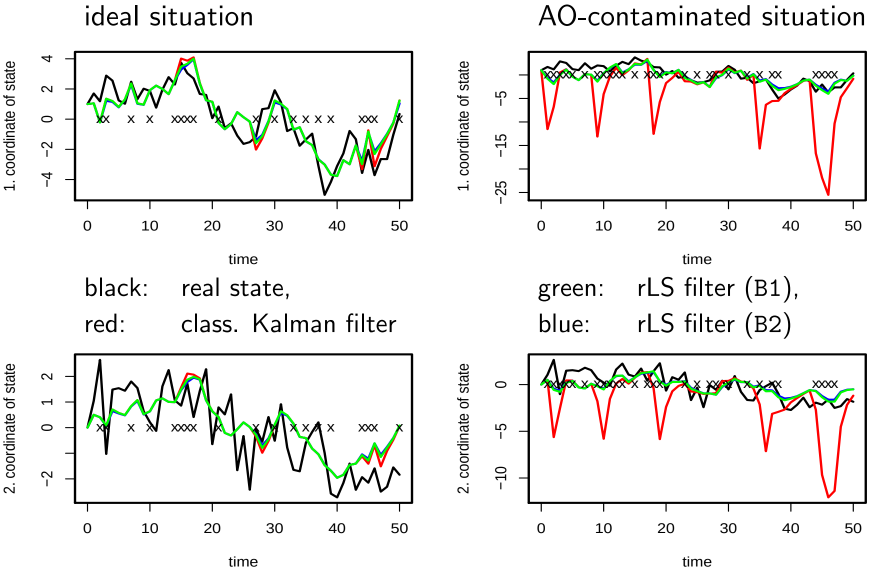
```
## specification of SSM: (p=2, q=1)
a0   ← c(1, 0); S0  ← matrix(0, 2, 2)
F    ← matrix(c(.7, 0.5, 0.2, 0), 2, 2)
Q    ← matrix(c(2, 0.5, 0.5, 1), 2, 2)
Z    ← matrix(c(1, −0.5), 1, 2)
Vi   ← 1;
## time horizon:
TT←50
## AO−contamination
mc   ← −20; Vc  ← 0.1; ract ← 0.1
## for calibration
r1←0.1; eff1←0.9

#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)
```

# Demonstration: rLSfilter II

```
### calibration b
#limiting S_{t|t-1}
SS <- limitS(S, F, Q, Z, Vi)
# by efficiency in the ideal model
(B1 <- rLScalibrateB(eff=eff1, S=SS, Z=Z, V=Vi))
# by contamination radius
(B2 <- rLScalibrateB(r=r1, S=SS, Z=Z, V=Vi))

### evaluation of rLS
rerg1.id <- rLSFilter(Yid, a, Ss, F, Q, Z, Vi, B1$b)
rerg1.re <- rLSFilter(Yre, a, Ss, F, Q, Z, Vi, B1$b)
rerg2.id <- rLSFilter(Yid, a, Ss, F, Q, Z, Vi, B2$b)
rerg2.re <- rLSFilter(Yre, a, Ss, F, Q, Z, Vi, B2$b)
```


ideal situation
AO-contaminated situation
1. coordinate of state
2. coordinate of state
time
black: real state,
red: class. Kalman filter
green: rLS filter (B1),
blue: rLS filter (B2)

# Bibliography


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

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

R Development Core Team (2006):
R:A language and environment for statistical computing.
R Foundation for Statistical Computing, Vienna, Austria.
`http://www.R-project.org`

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`

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