R–Package robKalman —

R. Kalman's revenge or Robustness for Kalman Filtering Revisited

Peter Ruckdeschel[1] Bernhard Spangl[2]

[1] Fraunhofer ITWM, Kaiserslautern, Germany, Peter.Ruckdeschel@itwm.fraunhofer.de
[2] Universität für Bodenkultur, Vienna, Austria, Bernhard.Spangl@boku.ac.at

Rennes, July 9, 2009

Euclidean State Space Models

Definitions and Assumptions:

— Time–Discrete, Euclidean Setup ideal model:

$$\begin{aligned} x_t &= F(x_{t-1},t)+v_t, & v_t &\overset{\text{indep.}}{\sim} (0,Q_t), & &[p\text{–dim}],\\ y_t &= Z(x_t,t)+\varepsilon_t, & \varepsilon_t &\overset{\text{indep.}}{\sim} (0,V_t), & &[q\text{–dim}],\\ & & x_0 &\sim (a_0,Q_0), & &[p\text{–dim}], \end{aligned}$$

$\{v_t\},\{\varepsilon_t\},x_0$ indep. as processes

functions F,Z smooth with known derivatives;
hyper–parameters Q_t,V_t,a_0 known

extensible to:

- continuous time (SDE's)
- incorporate user-specified controls

Types of Outliers

exogenous outliers affecting only singular observations

$$\begin{aligned} \text{AO} &\;::\; \varepsilon_t^{\text{re}} \sim (1-r_{\text{AO}})\mathcal{L}(\varepsilon_t^{\text{id}})+r_{\text{AO}}\mathcal{L}(\varepsilon_t^{\text{di}})\\ \text{SO} &\;::\; y_t^{\text{re}} \sim (1-r_{\text{SO}})\mathcal{L}(y_t^{\text{id}})+r_{\text{SO}}\mathcal{L}(y_t^{\text{di}}) \end{aligned}$$

endogenous outliers / structural changes

$$\text{IO} \;::\; v_t^{\text{re}} \sim (1-r_{\text{IO}})\mathcal{L}(v_t^{\text{id}})+r_{\text{IO}}\mathcal{L}(v_t^{\text{di}})$$

Notation: $\cdot \mathrel{\hat{=}} \cdot^{\text{id}}, \qquad \hat{\cdot} \mathrel{\hat{=}} \cdot^{\text{re}}, \qquad \tilde{\cdot} \mathrel{\hat{=}} \cdot^{\text{di}}$

Different and competing goals

A/SO attenuation of "false alarms"
IO tracking: detect structural changes as fast as possible;
recovering: clean data from structural changes
A/SO & IO identification problem:
simultaneous treatment only possible with delay

Classical Method: Kalman–Filter

Filter Problem

$$\mathrm{E}\left|x_t - f_t(y_{1:t})\right|^2 = \min_{f_t}!,$$

$$\text{with } y_{1:t}=(y_1,\ldots,y_t), \quad y_{1:0}:=\emptyset$$

General solution: $\mathrm{E}[x_t|y_{1:t}]$ —difficult to compute

Kalman–Filter

assuming $F(x,t)=F_t x$, $Z(x,t)=Z_t x$
optimal solution among linear filters — Kalman[/Bucy] [60/61]:

Initialization: $x_{0|0}=a_0$

Prediction: $x_{t|t-1}=F_t x_{t-1|t-1}, \qquad [\Delta x_t = x_t - x_{t|t-1}]$

Correction: $x_{t|t}=x_{t|t-1}+M_t^0\Delta y_t, \qquad [\Delta y_t = y_t - Z_t x_{t|t-1}]$

and corresponding recursions for the prediction/filtering error covariances $\Sigma_{t|t[-1]}$ and the Kalman gain M_t^0

Features of the Kalman–Filter

+ an easy, understandable structure: initialization, prediction, correction step
+ correction step is easily evaluable and interpretable: it is linear !
+ strict recursivity / Markovian structure: all information from the past useful for the future is captured in the value of $x_{t|t-1}$.

– the correction step is linear and thus not robust, as y enters unbounded;

Aim of robustification: try to retain all “+”’s, revise “−”

R-package `robKalman` — Contents

- Kalman filter: filter, Kalman gain, covariances
- ACM-filter: filter, multivariate version, GM-estimator
- rLS-filter: filter, calibration of clipping height
  - AO/SO-robust version
  - IO-robust version
  - with a certain delay joint treatment of AO/SO’s & IO’s
- extensible to further recursive filters:
  ⇝ general interface recursiveFilter
  with arguments:
  - data
  - state space model (hyper parameters)
    [will be: object of class SSM]
  - **functions for the init./pred./corr.step**
    [will be: object containing them]
  - [will be: control object]

Implementation concept

- Programming language
  - completely in `S`, perhaps some code in `C` later (⇝ `FKF`)
- Use existing infrastructure: `zoo`, `timeSeries`
  - for: graphics, diagnostics, management of date/time
- Code in different layers
  - internal functions: no `S4`-objects, no time stamps
    (helps bringing in code by “non-`S4`-people”)
  - user interface: `S4`-objects, time stamps
- Use generating functions for encapsulation
  - without using structured arguments:
    - too many arguments ⇝ user looses track
    - prone to name mis-matchings (positional, partial matching)
    - bad alternative: fix defaults. . .
  - have generating functions to produce control objects
  - control objects may be reused

Implementation so far

Interfaces so far

- preliminary, “`S4`-free” interfaces
  - Kalman filter (in our context) KalmanFilter
  - rLS: rLSFilter (=rLS.AO.Filter), rLS.IO.Filter, rLS.IOAO.Filter
  - ACM: ACMfilt, ACMfilter, mACMfilter
  - all realized as wrappers to recursiveFilter
- availability: `robKalman` version `0.3` (incl. demos)

`http://r-forge.r-project.org/projects/robkalman/`

Almost ready:

- `S4` classes: for SSM’s; for output-classes; for method-classes; for control-classes (reuse `robustbase`-code)
- interfaces between `S4`-layer and `S4`-free layer
  to other SSM packages
  to `robfilter` (**Roland Fried** & **K. Schettlinger**)

Work in process

Release Plans

- package `robKalman` should be on `CRAN` by UseR! 2009, but…
- at least: release on `CRAN` by end of August
- till then: refer to `r-forge`

Extensions

- robust smoothing (**80**% done)
- robust EM-Algorithm to estimate unknown hyper parameters (extending Shumway/Stoffer) (**70**% done)
- interpretation as random coefficient regression ⇝ robust regression-type approach (**rIC**, **mIC**) (**30**% done)
- connecttion to particle filters — theory and computer interface (**10**% done)
- speeding up things / bridging to fast Kalman filter of `FKF` by **David Luethi**, **Philipp Erb** (**1**% done)

Some experiences on collaborative programming on `r-forge`

- `r-forge`:
  very neat for collaborative R package development
  - version management (`svn`)
  - mail-forwarded `log`-files of committed code
    ⇝ keep track of work of others
  - bug tracker, archived mailing lists, . . .
  - see slides by **Stefan Theussl**
- needs serious conceptional preparations
  - for separating/modularizing tasks
  - consistency: coding & documentation conventions
- helpful: scheduling, reminders/deadlines for collaborators. . .
- summarizing:

Collaborative programming is enjoyable and very exciting!

THANKS FOR YOUR ATTENTION!

References

Birmiwal, K. and Shen, J. (1993) : *Optimal robust filtering.* Stat. Decis., **11**(2): 101–119.

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

Fried, R. and Schettlinger, K. (2008) : R-package `robfilter`: Robust Time Series Filters. `http://cran.r-project.org/web/packages/robfilter`.

Kalman, R.E. (1960) : A new approach to linear filtering and prediction problems. *Journal of Basic Engineering—Transactions of the ASME*, **82**: 35–45.

Kalman, R.E. and Bucy, R. (1961) : New results in filtering and prediction theory. *Journal of Basic Engineering—Transactions of the ASME*, **83**: 95–108.

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

Masreliez C.J. and Martin R. (1977) : Robust Bayesian estimation for the linear model and robustifying the Kalman filter. *IEEE Trans. Autom. Control*, **AC-22**: 361–371.

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

References (cont.)

R Development Core Team (2009) : R: *A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org`

R-Forge Administration and Development Team (2008) : *R-Forge User's Manual,* BETA. `SVN` revision: 47, August, 12 2008. `http://r-forge.r-project.org/R-Forge_Manual.pdf`

Schick, I.C. (1989) : *Robust recursive estimation of a discrete–time stochastic linear dynamic system in the presence of heavy-tailed observation noise*. Dissertation, Massachusetts Institute of Technology, Cambridge, MA.

Schick I.C. and Mitter S.K. (1994) : Robust recursive estimation in the presence of heavy-tailed observation noise. *Ann. Stat.*, **22**(2): 1045–1080.

Shumway, R.H. and Stoffer, D.S. (1982) : *An approach to time series smoothing and forecasting using the EM algorithm.* Journal of Time Series Analysis, **3**: 253–264.

Spangl, B. (2008) : *On Robust Spectral Density Estimation*. PhD Thesis at Technical University, Vienna.