Robust Kalman filter

Kjartan Halvorsen

2019-11-13

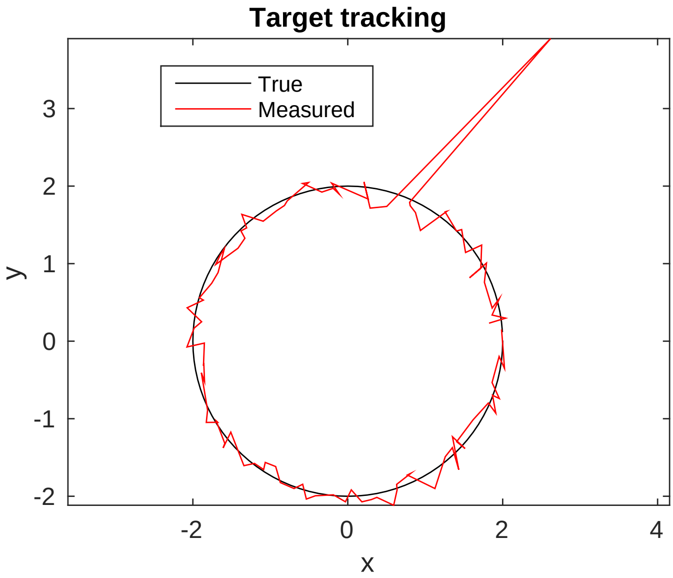
Why a robust version of the Kalman filter?

Why a robust version of the Kalman filter?

The Kalman filter assumes Gaussian measurement noise and so it is very sensitive to outliers.

Example 1

The target moves in a circle. Observations are noisy with one outlier



Example 1 contd.

The model of the dynamics: *Nearly constant velocity model*

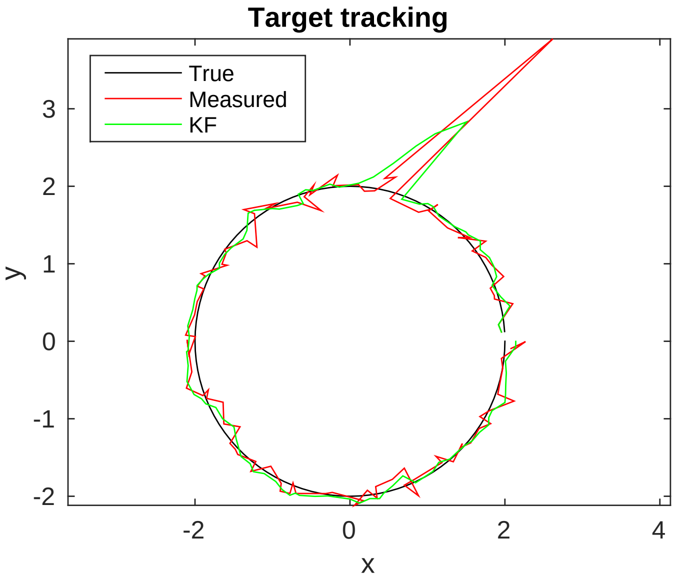
$$x(k+1) = \begin{bmatrix} I & hI \\ 0 & I \end{bmatrix} x(k) + \begin{bmatrix} \frac{h^2}{2} I \\ hI \end{bmatrix} v(k),$$

where the state vector contains the position and velocity of the target

$$x = \begin{bmatrix} p \\ \dot{p} \end{bmatrix}.$$

Example 1 contd.

Result of tracking using standard Kalman filter



Recommended reading

Mattingley, Jacob, and Stephen Boyd. "Real-time convex optimization in signal processing." Signal Processing Magazine, IEEE 27.3 (2010): 50-61.

Preperation exercise

Linear regression model

$$y(k) = ax(k) + b + e(k) + w(k),$$

where $e(k)$ is Gaussian noise and $w(k)$ is a sparse vector of outliers.

Preparation exercise, contd

Least squares estimation:

$$\text{minimize } ||y - ax - b||_2$$

Or, equivalently

$$\begin{aligned} &\text{minimize } ||\epsilon||_2 \\ &\text{subject to } \epsilon = y - ax - b \end{aligned}$$

Preparation exercise, contd

Least squares estimation:

$$\text{minimize } ||y - ax - b||_2$$

Solved by forming

$$A = \begin{bmatrix} x(1) & 1 \\ x(2) & 1 \\ \vdots & \vdots \\ x(N) & 1 \end{bmatrix}$$

and

$$z = \begin{bmatrix} a \\ b \end{bmatrix},$$

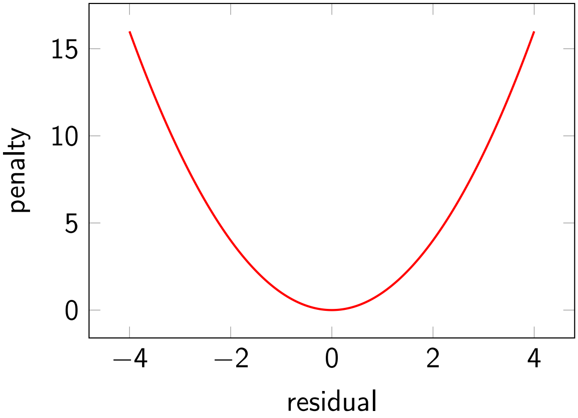
and solving for z in the (over-determined) system of equations

$$Az = y.$$

The problem with least squares

$$\text{minimize} \sum_k \phi_S(\epsilon_k)$$

where $\phi_S(u) = u^2$

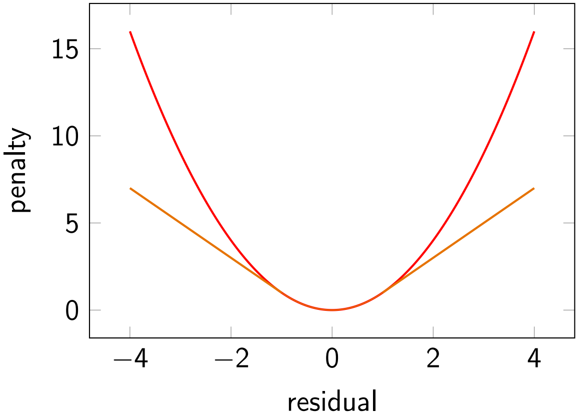


More robust: The Huber penalty function

A.k.a *robust least squares*

$$\text{minimize} \sum_k \phi_{hub}(\epsilon_k)$$

where $phi_{hub}(u) = \begin{cases} u^2 & |u| \le M \\ M(2|u| - M) & |u| > M \end{cases}$



Preparation exercis contd.

l1-regularization:

$$\text{minimize } ||y - ax - b - w||_2 + \gamma ||w||_1$$

Solved by convex optimization.

The vector w will contain the outliers. The larger the value of γ, the fewer non-zero elements in w.

The update step of the Kalman filter

We have the state space model

$$\begin{aligned} x(k+1) &= Hx(k) + Fv(k) \\ y(k) &= Cx(k) + w(k) + z(k) \end{aligned}$$

where

$$\begin{aligned} w &\sim \mathcal{N}(0, R) \\ v &\sim \mathcal{N}(0, Q) \end{aligned}$$

The measurement update of the Kalman filter can be shown to be equivalent to solving the problem

$$\begin{aligned} \text{minimize } & w^{\mathrm{T}} R^{-1} w + (x - \hat{x}_{k|k-1}) P^{-1} (x - \hat{x}_{k|k-1}) \\ \text{subject to } & \quad y = Cx + w \end{aligned}$$

with variables w and x.

Robust update

The idea is to write the update step using l1-regularization:

$$\text{minimize } w^{\mathrm{T}} R^{-1} w + (x - \hat{x}_{k|k-1}) P^{-1} (x - \hat{x}_{k|k-1}) + \lambda ||z||_1$$

$$\text{subject to} \quad y = Cx + w + z$$

with variables w, x and z. The matrix P is the covariance of the prediction error

$$P = P_{k|k-1} = \mathrm{E}(x - \hat{x}_{k|k-1})(x - \hat{x}_{k|k-1})^{\mathrm{T}}.$$

The parameter λ is tuned so that z has desired sparsity.

Robust update alternative form

The minization problem of the previous slide can be shown (next slide) to be equivalent to the problem

$$\text{minimize } (e - z)^{\mathrm{T}} S(e - z) + \lambda ||z||_1$$

with variable z. To compute S, first compute the Kalman gain

$$K = PC^{\mathrm{T}}(CPC^{\mathrm{T}} + R)^{-1},$$

and then

$$S = (I - CK)^{\mathrm{T}} R^{-1}(I - CK) + K^{\mathrm{T}} P^{-1} K.$$

The update is finally computed as

$$x = \hat{x}_{k|k-1} + K(e - z)$$

Obtaining the alternative form

Start with the criterion

$$\text{minimize } w^{\mathrm{T}} R^{-1} w + (x - \hat{x}_{k|k-1}) P^{-1} (x - \hat{x}_{k|k-1}) + \lambda ||z||_1.$$

Substitute

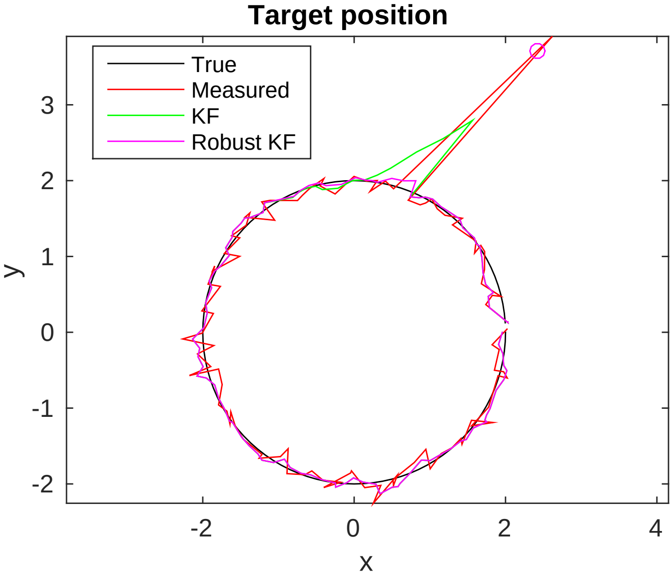
$$x = \hat{x}_{k|k-1} + K(e - z),$$
$$w = y - Cx - z$$

and use the identity

$$e = y - C\hat{x}_{k|k-1}.$$

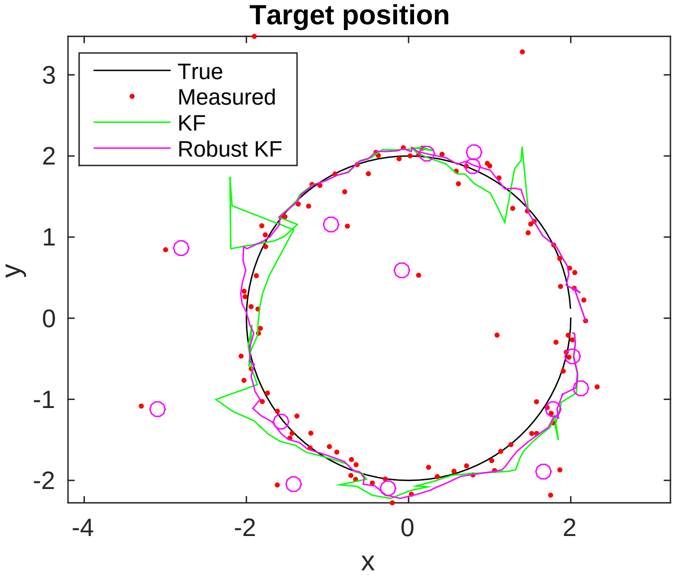
The alternative form follows.

Tracking example again



Tracking example again

10% chance of outlier
with 10 times normal
standard deviation



A fast and approximate implementation

The optimization problem is

$$\text{minimize } 0.5(e-z)^{\mathrm{T}}S(e-z)+\lambda||z||_1.$$

If S is diagonal, then we can assume the elements of e and z to have the same sign. The criterion can then be written

$$\text{minimize } 0.5(e-z)^{\mathrm{T}}S(e-z)+\lambda \mathrm{sign}(e)^{\mathrm{T}}z.$$

Expanding the quadratic form leads to

$$\text{minimize } 0.5e^{\mathrm{T}}Se - e^{\mathrm{T}}Sz + 0.5z^{\mathrm{T}}Sz + D^{\mathrm{T}}z$$

$$\Rightarrow \text{ minimize } 0.5z^{\mathrm{T}}Sz + C^{\mathrm{T}}z = f$$

Which has the solution obtained by setting the derivative of f wrt to z to zero:

$$df/dz = Sz + C = 0$$

hence

$$z = -S^{-1}C = e - \lambda S^{-1}\mathrm{sign}(e).$$

A fast and approximate implementation, contd

Note that we had assumed that the corresponding elements of z and e had the same sign. So, we need to check that this is the case and set to zero those elements of z that do not fulfill this requirement.

The method is only guaranteed to work for diagonal S. If S is not diagonal, an approximate solution can be found by forcing it to be diagonal. The inverse is then trivial to compute.

A fast and approximate implementation, contd

Matlab code

```
% Compute weighting matrix
% Have Kalman gain K, pred covariance Pkk
% and innovations ek = y - xk1
ICK = eye(m)-C*K;
S = ICK' / R * ICK + K' / Pkk * K;
% Works only if S is diagonal, so lets force it
% We will need the inverse only
Sinv = diag(1.0./diag(S));
se = sign(ek);
z = ek - lambda*Sinv*se;
z(find(sign(z) ~= se)) = 0;
% Filter update
xkNew = xk1 + K*(ek - z);
```