The Parameter Optimization of Kalman Filter Based on Multi-Objective Memetic Algorithm

Yu-Dan Huo
Department of Computer Science
China University of Geosciences
Wuhan, China
yudanhuo@163.com

Zhi-Hua Cai
Department of Computer Science
China University of Geosciences
Wuhan, China
zhcai@cug.edu.cn

Wen-Yin Gong
Department of Computer Science
China University of Geosciences
Wuhan, China
wygong@cug.edu.cn

Qin Liu
Department of Computer Science
China University of Geosciences
Wuhan, China
lq8738@126.com

ABSTRACT

Generally, there are two objectives in the optimization of the measurement noise covariance matrix R of Kalman filter. However, most of the traditional optimization methods of Kalman filter only focus on one objective. In this paper, we proposed a new method to optimize the parameter R based on Multi-Objective Memetic Algorithm (MOMA). Compared with traditional methods, it can optimize multiple objectives simultaneously. In this method, the decision vector is the diagonal elements of matrix R, the first objective function f_1 is the mean of the residual vectors, and the second objective function f_2 is the degree of mismatching between the actual value of the residual covariance with its theoretical value. In the MOMA, the global search based on NSGA-II is utilized to minimize the two objective functions, and the local search based on Simulated Annealing (SA) is just used to minimize the f_1. The experimental results demonstrate that the Kalman filter optimized by MOMA, namely MOMA-Kalman, can get much smaller filtering error than regular Kalman filter and other adaptive filter algorithms, such as SageHusa-Kalman and Fuzzy-Kalman.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*global optimization*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*control theory*

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.

http://dx.doi.org/10.1145/2576768.2598242.

Keywords

Kalman filter; parameter optimization; multi-objective optimization; memetic algorithm

1. INTRODUCTION

Kalman Filter (KF) [1] is one of the well-known and efficient methods for estimating the state of dynamic system from an incomplete and noisy measurement. Since it was proposed in 1960s, Kalman filter has been widely applied in many fields, such as navigation, signal processing, control system and information fusion. It also has many improved variants like Extended Kalman Filter (EKF) [2] and Unscented Kalman Filter (UKF) [3] which are utilized in nonlinear system. The Kalman filter works well in the condition that the *a priori* statistics of the stochastic errors in both dynamic process and measurement models are assumed to be available, which is very difficult in practical applications, especially the measurement noise covariance matrix R. To solve this problem, many adaptive mechanisms are used into Kalman filter, which is called Adaptive Kalman Filter (AKF). According to the filtering performance, adaptive Kalman filter can optimize or estimate its noise statistics parameters adaptively to adjust to the change of process or measurement noise. Mehra [4] classified the different methods of adaptive filter into four categories: Bayesian, maximum likelihood, correlation and covariance matching.

The traditional adaptive Kalman filter algorithms include Sage-Husa Kalman [5] based on maximum *a posteriori* probability estimation, Fuzzy Kalman [6] based on fuzzy logic and covariance matching, Bayes adaptive filter [7] based on Bayesian estimation, Robust Kalman [8] and so on. In recent years, more and more people start to use Evolutionary Algorithms (EA) into adaptive Kalman filter. In literature, Szabat [9], Salvatore [10], Jatoth [11], and Mosavi [12] published some of the earlier work on using evolutionary algorithms to optimize the initial parameter value of Kalman filter. Generally, there are two objectives in the parameter optimization of Kalman filter. One is to reduce the filtering error ,and the other one is to estimate the actual value of the parameters. However, all the adaptive Kalman filter methods mentioned above only focus on one objective. For example, Sage-Husa Kalman focuses on the error reduction and Fuzzy Kalman focuses on the actual parameter estimation. Even the adaptive Kalman filter based on single-objective evolutionary algorithm, just can optimize one objective each time.

In this paper, we proposed to utilize multi-objective memetic algorithm (MOMA) to optimize the parameter R of Kalman filter, which can optimize the two objectives mentioned above simultaneously. MOMA [13, 14] is the multi-objective optimization version of memetic algorithm (MA) [15]. It has been successfully applied to many fields, such as knapsack problems [16], dynamic location problems [17], art classifiers [18], transmission network expansion planning [19] and environmental power unit commitment [20]. The MOMA used in this paper is a combination of NSGA-II [21] and Simulated Annealing (SA) [22]. In the optimization, The first objective function f_1 is the mean value of the residual vectors of Kalman filter. In certain extend it symbolizes the filtering error. The second objective function f_2 is the degree of mismatching between the actual value of the residual covariance with its theoretical value. It can be used to find the actual value of R. For the purpose of getting smaller error under the premise that the optimized value of R is close to its actual value, the local search based on SA just minimizes the first objective function.

The rest of this paper is organized as follows: Section II introduces the Kalman filter and its parameter optimization problem. Section III introduces the new method of optimizing the parameter R of Kalman filter based on multi-objective memetic algorithm. Some simulation results are presented in Section IV to show its performance. Finally, Section V concludes the paper.

2. PARAMETER OPTIMIZATION OF KALMAN FILTER

2.1 Kalman Filter

Kalman filter is one of the most popular algorithms in the control area. It is always used to estimate the state of a dynamic system. The system model and measurement model for a simple linear discrete-time Kalman filter are represented as:

$$x_k = \Phi x_{k-1} + \omega_k \tag{1}$$

$$z_k = H x_k + \nu_k \tag{2}$$

where $x_k \in R^n$ is the system state vector, $\omega_k \in R^n$ is the system noise vector, $z_k \in R^m$ is the measurement vector to system state, and $\nu_k \in R^m$ is the measurement noise vector. Φ is the state transition matrix, which reflects the mathematical or physical relationship between system state x_k and x_{k-1}. H is the measurement matrix, which represents the relationship between the measurement z_k and system state x_k. The vector ω_k and ν_k are both white noise sequences with zero means and mutually independent

$$E[\omega_k \omega_i^T] = \begin{cases} Q, & i = k \\ 0, & i \neq k \end{cases} ; \tag{3}$$

$$E[\nu_k \nu_i^T] = \begin{cases} R, & i = k \\ 0, & i \neq k \end{cases} ; \tag{4}$$

$$E[\omega_k \nu_i^T] = 0, \text{for all } i \text{ and } k. \tag{5}$$

where non-negative definite matrix Q is the system noise covariance matrix, positive definite matrix R is the measurement noise covariance matrix, $E[\bullet]$ represents expectation, and superscript "T" denotes matrix transpose.

The purpose of Kalman filter is to estimate the actual value of x_k in equation (1). Based on the model equations (1)-(5), the key five equations of discrete-time Kalman filter is summarized as follows:

$$\hat{x}_k^- = \Phi \hat{x}_{k-1}; \tag{6}$$

$$P_k^- = \Phi P_{k-1} \Phi^T + Q; \tag{7}$$

$$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1}; \tag{8}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - H \hat{x}_k^-); \tag{9}$$

$$P_k = (I - K_k H) P_k^-. \tag{10}$$

Equations (6)-(7) are the time update equations of Kalman filter from step $k-1$ to k. These equations generate *a priori* estimation of system state at step k. Equations (8)-(10) are the measurement update equations of the algorithm. They incorporate the measurement value z_k into *a priori* estimation to obtain an improved *a posteriori* estimation, which is the output of Kalman filter at step k. In the above equations, $\hat{x}_k$ is the estimation value of the system state x_k, P_k is the error covariance matrix defined by $E[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T]$, and weighting matrix K_k is the Kalman gain matrix. The Kalman filter algorithm starts with an initial condition value $\hat{x}_0$ and P_0.

The procedure of Kalman filter algorithm is showed by Algorithm 1.

Algorithm 1 Kalman Filter

1: Set the parameters Φ, H, Q and R;
2: Initialize the $\hat{x}_0$, P_0, $k = 1$;
3: **while** (need to estimate the system state) **do**
4: Time Update:
$\hat{x}_k^- = \Phi \hat{x}_{k-1}$;
$P_k^- = \Phi P_{k-1} \Phi^T + Q$;
5: Get the measurement z_k;
6: Measurement Update:
$K_k = P_k^- H^T (H P_k^- H^T + R)^{-1}$;
$\hat{x}_k = \hat{x}_k^- + K_k (z_k - H \hat{x}_k^-)$;
$P_k = (I - K_k H) P_k^-$;
7: $k = k + 1$;
8: **end while**

2.2 Parameter Optimization of Kalman Filter

Kalman filter is a very powerful method to estimate the system state. But it only works well in the condition that the parameters Φ, H, Q and R in the equations (6)-(10) are precisely known. Inaccurate values of these parameters will reduce the filtering accuracy, increase the filtering error, and even cause filter divergence. There have been many works on optimizing these parameters.

We can get the Φ and H by building accurate system and measurement models. The value of Q is generally stable in a system, although there is no direct method to get it. The most difficult, and also important is to estimate the value of R, because of its variability. For example in the navigation system, one very important application of Kalman filter, once the external environment of the target which is being navigated has changed, the value of R will change immediately. So a multi-objective memetic algorithm (MOMA) is used in this paper to optimize the parameter R of Kalman filter. It can minimize two objective functions simultaneously, compared to the other methods which can just minimize only one objective function each time.

2.3 Selection of Objective Functions

In the optimization of parameter R, generally there are two objectives. The first one is to decrease the root mean square error (RMSE) of filtering. We use the mean of residual vectors as the first objective function [23]. For Kalman filter, at filter step k, the residual ε_k is defined as follows:

$$\varepsilon_k = \tilde{z}_k = z_k - \hat{z}_k^- = z_k - H\hat{x}_k^- \tag{11}$$

where z_k is the measurement at step k, H is the measurement matrix, $\hat{x}_k^-$ is the predictive value of state vector x_k, and $\hat{z}_k^-$ defined by $H\hat{x}_k^-$ is the estimation of measurement z_k. From the definition we can see that the ε_k includes the new information from measurement z_k. From equation (9), if $\varepsilon_k = 0$, then the predictive value of state vector x_k is the estimation value. That means the estimation at step $k-1$ is very accurate. So the residual ε_k symbolizes the filter error in certain extend. Then we can define the first objective function as follows:

$$f_1(x) = \frac{1}{S}\sum_{k=1}^{S} \varepsilon_k^T \varepsilon_k \tag{12}$$

where S is the total step number of filtering.

The second objective is to estimate the actual value of R. Based on covariance-matching techniques [24], we designed the second objective function. The basic idea behind the covariance-matching techniques is to make the actual value of the covariance of the residual ε_k consistent with its theoretical value. From Kalman filter equations, the theoretical covariance of residual ε_k is

$$C_t = HP_k^- H^T + R. \tag{13}$$

The actual covariance of residual ε_k is approximated by its sample covariance

$$C_a = \frac{1}{M}\sum_{i=1}^{M} \varepsilon_i \varepsilon_i^T \tag{14}$$

where M is the window size which is chosen empirically to give some statistical smoothing. When filtering, if the value of R is accurate, C_t and C_a are basically the same. So we can define the second objective function based on the degree of mismatching between C_t and C_a:

$$f_2(x) = \left| \frac{trace(C_a)}{trace(C_t)} - 1 \right| \tag{15}$$

where $trace(\bullet)$ represents calculating the trace of the matrix in the bracket. The closer the estimation of parameter R to its actual value, the closer the value of $f_2(x)$ to zero.

The function $f_1(x)$ defined in equation (12) and the function $f_2(x)$ defined in equation (15) are the objective functions we used in the multi-objective memetic algorithm to optimize the parameter R of Kalman filter.

3. OPTIMIZATION OF R BASED ON MULTI-OBJECTIVE MEMETIC ALGORITHM

3.1 Multi-Objective Optimization Model

Without loss of the generality, a multi-objective optimization problem with a set of m decision variables, a set of n constrains and a set of k objective functions can be described as follow:

$$\begin{gathered} minimize : y = f(x) = (f_1(x), ..., f_k(x)) \\ subject\ to : e(x) = (e_1(x), ..., e_n(x)) \leq 0 \end{gathered}$$

where $x = (x_1, x_2, ..., x_m) \in D$ is the decision vector, $y = (y_1, y_2, ..., y_k) \in Y$ is the objective vector, D denotes as the decision space, and Y means the objective space. Generally, for each decision vector, it satisfies the constrain functions. In most instances, there are contradictions between objective functions. The reduction of one objective function may cause a increase of another objective function. And generally, it's impossible to make all the objective functions to reach the optimal value.

In a certain sense, the problem of optimizing parameter R is a kind of multi-objective optimization problem. R is a positive definite matrix, so only the diagonal elements need to be optimized. Assume R is a m-dimensional square matrix, so the decision vector can be assumed as $\mathbf{x} = (r_{11}, r_{22}, ..., r_{mm})$, where $r_{ii}(i = 1, \ldots, m)$ is the i-th diagonal element of matrix R. Then this optimization problem can be abstracted to a multi-objective optimization problem:

$$\begin{gathered} minimize : y = f(x) = (f_1(x), f_2(x)) \\ subject\ to : minR \leq r_{ii} \leq maxR, i = 1, 2, \ldots, m \\ where : x = (r_{11}, r_{22}, ..., r_{mm)} \end{gathered}$$

where $minR$ and $maxR$ are the lower limit and upper limit to the elements of R, objective functions $f_1(x)$ and $f_2(x)$ are defined in equation (12) and (15). In general, one possible value of R can't make both the $f_1(x)$ and $f_2(x)$ to reach the minimum.

3.2 Memetic Algorithm

Memetic algorithm (MA), inspired by both Darwinian principle of natural evolution and Dawkins' notion of cultural evolution, is a metaheuristic search method [25]. In general, MA can be viewed as a combination of a population-based global optimization technique and a individual-based local heuristic search method, which improves the capability of global optimization algorithms like GA for finding optimal solutions in the optimization problems with higher convergence speed. Like GA, memetic algorithm is also a population-based algorithm. Its unique aspect is that the individuals are promoted to get some experiences from the others in the population by a local search process between traditional evolutionary process [25]. In some problem domains, MA has been shown to be both more efficient and more effective than traditional evolutionary algorithm [26]. In a MA, the initial population is generated randomly, and then the fitness of every individual is improved by local search. In each generation, original genetic operators, such as selection, crossover and mutation, are applied like in the GA, and then, local search is executed again to improve the population quality. Algorithm 2 explains the procedure of a simple single-objective memetic algorithm:

Algorithm 2 Simple Memetic Algorithm

t:=0;
$P(t)$:=initPop();
$P(t)$:=localSearch($P(t)$);
evaluateFitness($P(t)$);
while (stoping criteria not met) **do**
    $P^{'}(t)$:=selectForVariation($P(t)$);
    $P^{'}(t)$:=recombine($P^{'}(t)$);
    $P^{'}(t)$:=mutate($P^{'}(t)$);
    $P^{'}(t)$:=localSearch($P^{'}(t)$);
    evaluateFitness($P^{'}(t)$);
    $P(t+1)$:=selectNewPop($P(t)$,$P^{'}(t)$);
    t:=t+1;
end while

3.3 Multi-Objective Memetic Algorithm (MOMA)

In the optimization of parameter R, when the two objective functions are minimized simultaneously, we hope to reduce the filtering error as much as we can. So the objective function $f_1(x)$ should get more attention in the optimization process. Multi-objective memetic algorithm (MOMA) is a perfect choice for this purpose. Using MOMA, we can minimize the function $f_1(x)$ and $f_2(x)$ simultaneously at the global search, and then only minimize $f_1(x)$ at the local search. The MOMA we used in this paper is a combination of NSGA-II and Simulated Annealing (SA). NSGA-II is used for global search and SA is used for local search.

Proposed by Prof. Kalyanmoy Deb [21], NSGA-II is a very famous algorithm for solving multi-objective optimization problems. In the fast non-dominated sort of NSGA-II, the individuals are sorted into each front. The individuals in the first front are completely non-dominant and the ones in the second front are dominated by the individuals in the first front only and the front goes so on. Each individual are assigned rank values as the fitness. The rank value of the individuals in first front is 1 and the rank value of the individuals in second front is 2 and so on. Besides of the rank value, crowding distance is calculated for each individual to measure how close it is to its neighbors. The mechanism of crowding distance is utilized to improve the diversity of the solutions. In each generation, tournament selection is used to select parents based on the rank value and crowding distance. And then, crossover and mutation operators are utilized on parents to generate offsprings. At last, merger the parents and offsprings, and select the best N individuals based on rank value and crowding distance to generate the population for the next generation, where N is the size of population. The pseudo-code of MOMA based on NSGA-II is showed by Algorithm 3.

Simulated Annealing (SA) [22] is utilized in the MOMA proposed for local search. SA is a generic probabilistic metaheuristic for the global optimization problem. It comes from annealing in metallurgy, a approach to heat and control cooling of a material to increase the size of its crystals and reduce their defects. It has been proven that, for any given finite problem, the probability that the SA get the global optimal solution approaches 1 if there is enough annealing time. The SA used in MOMA is showed by Algorithm 4.

Algorithm 3 Multi-Objective Memetic Algorithm

Initialize the population Pop randomly;
Local search by Simulated Annealing on Pop;
Sort Pop by non-domination sort and crowding distance;
while (stopping criteria not met) **do**
    Generate parents $P \leftarrow$ tournament selection on Pop;
    Generate offspring $O \leftarrow$ genetic operator on P;
    Local search by Simulated Annealing on O every five generations;
    Generate intermediate population $I \leftarrow P \cup O$;
    Sort I by non-domination sort and crowding distance;
    $Pop \leftarrow$ select previous $sizePop$ individuals of I;
end while

Algorithm 4 Simulated Annealing on Pop

Initialize $T_0 = 10, kmax = 8, ratio = 0.9$;
for $i = 1$ to $sizePop \times 0.1$ **do**
    $x \leftarrow Pop(i), T \leftarrow 10^6, k \leftarrow 0$;
    while $k < kmax$ and $T > T_0$ **do**
        $x^{'} \leftarrow neighbour(x)$;
        $\Delta f_1 \leftarrow f_1(x^{'}) - f_1(x), \Delta f_2 \leftarrow f_2(x^{'}) - f_2(x)$;
        if $\Delta f_1 < 0$ and $\Delta f_2 < 0$ **then**
            $x \leftarrow x$';
        end if
        if $\Delta f_1 < 0$ and $\Delta f_2 \geq 0$ **then**
            if $e^{-\frac{\Delta f_2}{T}} \geq rand(1)$ **then**
                $x \leftarrow x$';
            end if
        end if
        $k \leftarrow k+1, T \leftarrow T * ratio$;
    end while
end for

In MOMA, the balance of global search and local search is a very important issue [27]. Considering both the simplicity and the balance of global search and local search, the SA is applied to only the 10 percent best individuals of each generation after every 5 generations of global search.

3.4 Optimization of R based on MOMA

The MOMA is an iterative algorithm and it spends a lot of time for running. However, Kalman filter is a real-time approach, and it has high demands on the running time. So the optimization of parameter R using MOMA must be in offline mode. That means the parameter R is optimized by MOMA at first, and then the optimal value is used in a regular Kalman filter.

The schematics of optimizing R in one generation is described by Figure. 1. In Figure. 1, there are N Kalman filters, representing the N individuals in a population with different values of parameter R. Using their own parameter R, the filters run on the historical data, and evaluate the objective functions f_1 and f_2 based on the filter results. Then the generation begins, including non-dominated sort, calculating crowding distance, selection, crossover, mutation and local search. After the generation, the old values of R are updated by new values. Repeat this process until the stopping criteria is reached. At last, N optimal values of R are obtained.

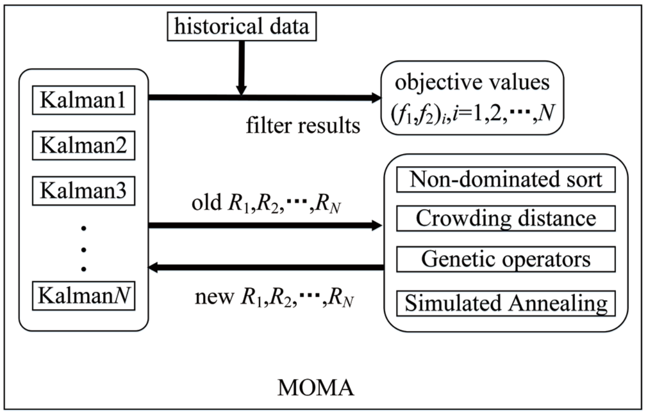

Figure 1: The process of optimizing parameter R in one generation.

4. SIMULATION EXPERIMENTS

In this section, simulation experiments have been carried out to compare the performance of MOMA-Kalman, whose parameter R is optimized by multi-objective memetic algorithm, with the other variants of Kalman filter.

4.1 Simulation Model Based on Target Tracking

The experiments are based on a simulation of rocket target tracking. Assume a rocket is doing uniformly accelerated motion escaping from the Earth. Its acceleration is $20m/s^2$. The fluctuations of engine thrust always cause some fluctuations of the acceleration. A radar on the ground is tracking the rocket and it gives the observation of the distance of the rocket from the ground every second. The observations are noisy. Now we need to estimate the displacement, velocity and acceleration of the rocket every second using Kalman filter.

So the system state at k-th second is $x_k = [d_k, v_k, a_k]^T$, where d_k indicates the displacement of the rocket at k-th second, v_k indicates the velocity and a_k indicates the acceleration. According to the physical model of uniformly accelerated motion, the system model is

$$x_k = \begin{bmatrix} 1 & 1 & 0.5 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \times x_{k-1} + \omega_{k-1} \quad (16)$$

where ω_{k-1} is zero-mean Gaussian white noise with covariance matrix $Q = diag([0, 0, 0.1])$ caused by engine thrust fluctuations. The measurement model is

$$z_k = [1 \ \ 0 \ \ 0] \times x_k + \nu_k \quad (17)$$

where z_k is the observation of the rocket displacement at k-th second and ν_k is the measurement noise with covariance matrix R. The initial conditions in the simulation are assumed to be $x_0 = [1000, 50, 20]^T$, $\hat{x}_0 = [990, 0, 0]^T$ and the initial error covariance matrix is given by $P_0 = diag([30, 20, 10])$. The rocket state is estimated by Kalman filter from 0 to 200s.

4.2 Experiment Results and Analysis

In practice, the value of R is unknown. Based on the measurement information, MOMA-Kalman uses the multi-objective memetic algorithm we proposed in this paper to optimize the R at first, and then uses the optimal value into regular Kalman filter to estimate the state of the rocket. Set the population size of MOMA is 20, and the pareto front of optimizing the parameter R we get is showed in Figure. 2. From Figure. 2 we can see that, the MOMA well minimized the objective function $f_1(x)$ and $f_2(x)$ simultaneously.

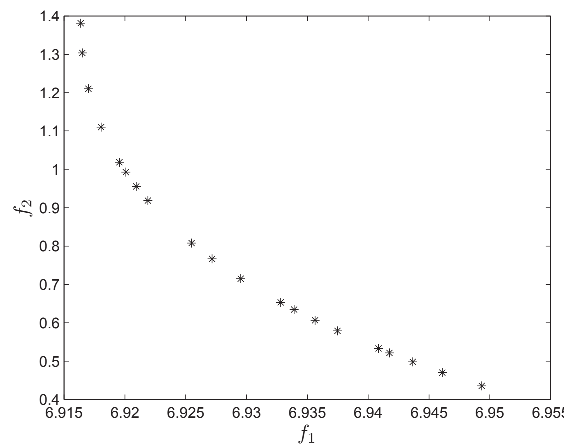

Figure 2: The pareto front of R obtained by MOMA.

The comparison results of the MOMA-Kalman with other Kalman filter algorithms on root mean square error(RMSE) of filtering are listed in Table 1, where Regular-Kalman represents the regular Kalman filter algorithm, SageHusa-Kalman [5] is an adaptive Kalman filter based on maximum *a posteriori* probability estimation, and Fuzzy-Kalman [6] is an adaptive Kalman filter based on fuzzy logic. In MOMA-Kalman, the initial value of R is optimized by MOMA, and in NSGA-Kalman, the initial value of R is optimized by NSGA-II. Each Kalman filter algorithm run 30 times, and each time Regular-Kalman, SageHusa-Kalman and Fuzzy-Kalman share a same initial value of R which is selected randomly, each time MOMA-Kalman and NSGA-Kalman stop the optimization until 500 times of evaluation. In the experiment, MOMA-Kalman and NSGA-Kalman chose the first individual of the pareto set as the optimal value of parameter R. However, in practical applications, the optimal value can be chosen by actual demand. If the first objective function is paid more attention on than the second one, then the individual which has the smallest value of $f_1(x)$ can be chosen as the final optimal value of R. Conversely, if the second objective function is paid more attention on, then the individual which has the smallest value of $f_2(x)$ is chosen.

In Table 1, the smallest error of each time is marked in bold. It is clear that, MOMA-Kalman got the smallest error 25 times in comparison. It has been proven that the method of optimizing the parameter R of Kalman filter by MOMA is very effective.

Table 1: The comparison results on root mean square error.

Run time	Regular-Kalman	SageHusa-Kalman	Fuzzy-Kalman	NSGA-Kalman	MOMA-Kalman
1	6.1378	6.3661	6.1053	5.2829	**5.2384**
2	5.8159	6.4088	5.7881	5.4019	**5.2855**
3	5.8083	6.4101	5.7809	5.2729	**5.2384**
4	7.0481	6.3114	7.0048	**5.2854**	**5.2854**
5	5.9855	6.3841	5.9533	**5.2384**	**5.2384**
6	5.9878	6.3838	5.9555	5.6242	**5.2384**
7	6.6822	6.3250	6.6350	5.2851	**5.2384**
8	6.4088	6.3418	6.3734	**5.2854**	5.2855
9	5.7981	6.4117	5.7711	5.2895	**5.2384**
10	7.1010	6.3101	7.0559	**5.2854**	**5.2854**
11	7.4249	6.3052	7.3691	**5.2384**	**5.2384**
12	6.8134	6.3191	6.7778	**5.2854**	**5.2854**
13	5.3235	6.5312	5.2737	**5.2853**	5.2854
14	6.0743	6.3732	6.0437	5.3629	**5.2863**
15	6.2531	6.3547	6.2176	5.3003	**5.2851**
16	6.9887	6.3130	6.9473	5.2855	**5.2854**
17	6.4267	6.3405	6.3906	5.3888	**5.2384**
18	5.2686	6.6102	**5.1176**	5.2855	5.2854
19	7.2080	6.3079	7.1595	5.3271	**5.2384**
20	6.8654	6.3171	6.8281	**5.2384**	**5.2384**
21	5.4331	6.6228	**5.2015**	5.2384	5.2384
22	7.4014	6.3054	7.3463	**5.2384**	**5.2384**
23	6.3580	6.3458	6.3185	5.3163	**5.2384**
24	5.3481	6.5214	5.3046	5.2869	**5.2384**
25	6.0218	6.3795	5.9930	**5.2853**	5.2854
26	7.1491	6.3090	7.1025	**5.2384**	**5.2384**
27	5.6307	6.4432	5.5969	5.2839	**5.2384**
28	7.0224	6.3121	6.9799	**5.2384**	**5.2384**
29	7.0324	6.3118	6.9896	**5.2384**	**5.2384**
30	7.5263	6.3046	7.4673	5.2602	**5.2383**
mean	6.4114	6.3760	6.3616	5.2958	**5.2556**
variance	4.777×10^{-1}	7.8×10^{-3}	4.940×10^{-1}	5.6×10^{-3}	$\mathbf{5.3250\times10^{-4}}$

The comparison results of MOMA-Kalman and Regular-Kalman at 22-th run are visualized in Figure. 3, with the x-coordinate being the filter time from 0 to 200s and y-coordinate being filter error. At k-th second, the filter error is defined by

$$Filter\ error = \sqrt{(\hat{d}_k - d_k)^2 + (\hat{v}_k - v_k)^2 + (\hat{a}_k - a_k)^2}$$

where d_k, v_k and a_k are the real value of the rocket displacement, velocity and acceleration at k-th second, $\hat{d}_k$, $\hat{v}_k$ and $\hat{a}_k$ are the estimated value obtained by Kalman filter. From Figure. 3, it is obvious that the MOMA-Kalman has a much greater convergence speed than Regular-Kalman.

The comparison results on estimating the real value of rocket displacement, velocity and acceleration are also shown in Figure. 4, Figure. 5 and Figure. 6. From these figures we can see, the estimates of MOMA-Kalman converge to the real value much faster than the estimates of Regular-Kalman.

5. CONCLUSIONS

A new method to optimize the measurement noise covariance matrix R of Kalman filter based on multi-objective memetic algorithm (MOMA) is developed in this paper. Compared with traditional optimization methods of Kalman filter, which only focus on one objective function in the process of optimization, this new method can optimize multiple objective functions simultaneously. The experimental results based on a simulation of target tracking demonstrated its effectiveness. The Kalman filter optimized by MOMA, namely MOMA-Kalman, can get much smaller filtering error than regular Kalman filter, SageHusa-Kalman, Fuzzy-Kalman and NSGA-Kalman.

6. ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China under Grant No.61203307, and

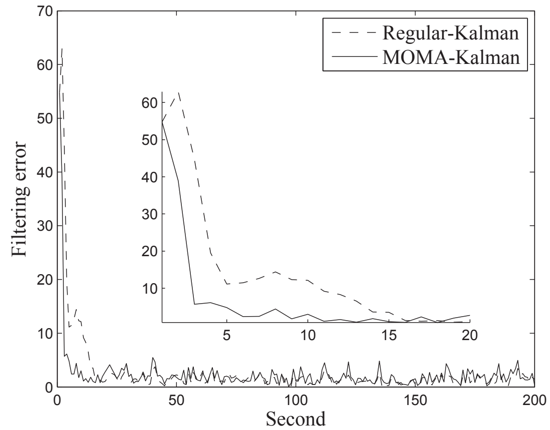

Figure 3: Comparison of MOMA-Kalman and Regular-Kalman on the filtering error at k-th second.

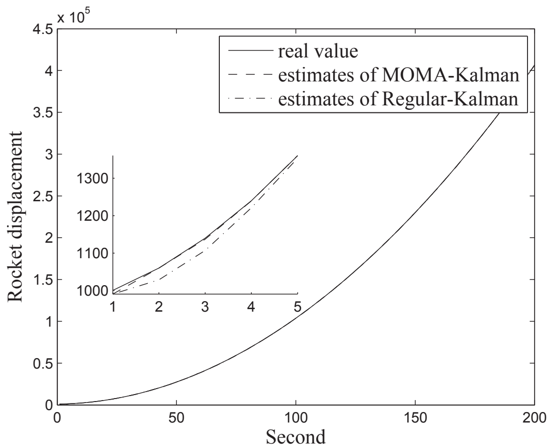

Figure 4: Comparison on estimating the rocket displacement.

the Natural Science Foundation of Hubei Province under Grant No. 2013CFA004.

7. REFERENCES

[1] R. E. Kalman et al. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

[2] R. Dhaouadi, N. Mohan, and L. Norum. Design and implementation of an extended kalman filter for the state estimation of a permanent magnet synchronous motor. *IEEE Transactions on Power Electronics*, 6(3):491–497, 1991.

[3] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte. A new method for the nonlinear transformation of means and covariances in filters and estimators. *IEEE Transactions on Automatic Control*, 45(3):477–482, 2000.

[4] R. Mehra. Approaches to adaptive filtering. *IEEE Transactions on Automatic Control*, 17(5):693–698, 1972.

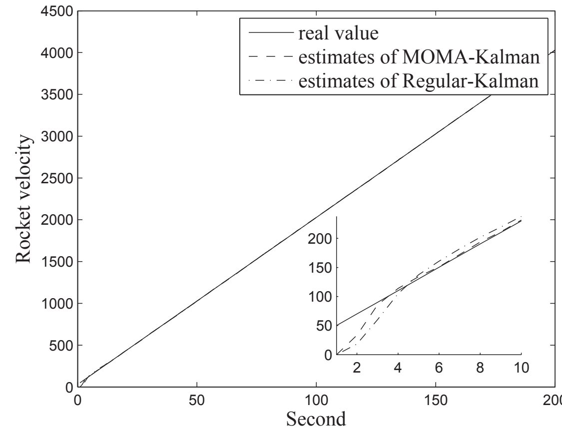

Figure 5: Comparison on estimating the rocket velocity.

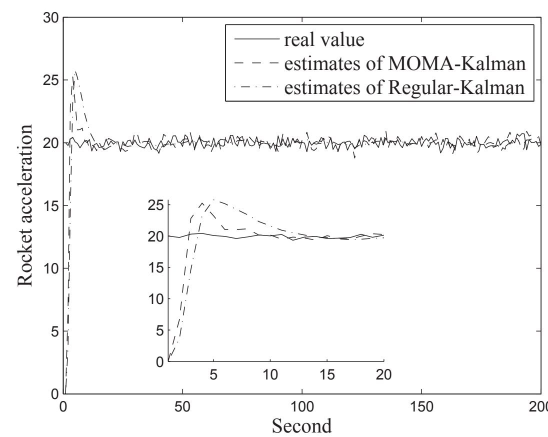

Figure 6: Comparison on estimating the rocket acceleration.

[5] M. Narasimhappa, P. Rangababu, S. L. Sabat, and J. Nayak. A modified sage-husa adaptive kalman filter for denoising fiber optic gyroscope signal. In *2012 Annual IEEE India Conference*, pages 1266–1271. IEEE, 2012.

[6] B. Talel and B. H. Faycal. Fuzzy kalman filter for non linear stochastic systems. In *10th International Multi-Conference on Systems, Signals & Devices*, pages 1–7. IEEE, 2013.

[7] S. Sarkka and J. Hartikainen. Non-linear noise adaptive kalman filtering via variational bayes. In *IEEE International Workshop on Machine Learning for Signal Processing*, pages 1–6. IEEE, 2013.

[8] X. M. Qu. Robust kalman track fusion in target tracking with uncertainties. In *International Conference on Computational Problem-Solving*, pages 1–4. IEEE, 2011.

[9] K. Szabat and T. Orlowska-Kowalska. Performance improvement of industrial drives with mechanical elasticity using nonlinear adaptive kalman filter. *IEEE*

Transactions on Industrial Electronics, 55(3):1075–1084, 2008.

[10] N. Salvatore, A. Caponio, F. Neri, S. Stasi, and G. L. Cascella. Optimization of delayed-state kalman-filter-based algorithm via differential evolution for sensorless control of induction motors. *IEEE Transactions on Industrial Electronics*, 57(1):385–394, 2010.

[11] R. K. Jatoth, D. N. Rao, and K. S. Kumar. Particle swarm optimization aided unscented kalman filter for ballistic target tracking. In *IEEE International Conference on Communication Control and Computing Technologies*, pages 455–460. IEEE, 2010.

[12] M. Mosavi, M. Sadeghian, and S. Saeidi. Increasing dgps navigation accuracy using kalman filter tuned by genetic algorithm. *International Journal of Computer Science Issues*, 8(6), 2011.

[13] J. D. Knowles and D. W. Corne. M-paes: A memetic algorithm for multiobjective optimization. In *2000 Congress on Evolutionary Computation*, volume 1, pages 325–332. IEEE, 2000.

[14] D. Liu, K. C. Tan, C. K. Goh, and W. K. Ho. A multiobjective memetic algorithm based on particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 37(1):42–50, 2007.

[15] X. S. Chen, Y. S. Ong, M. H. Lim, and K. C. Tan. A multi-facet survey on memetic computation. *IEEE Transactions on Evolutionary Computation*, 15(5):591–607, 2011.

[16] H. Ishibuchi and S. Kaige. Implementation of simple multiobjective memetic algorithms and its application to knapsack problems. *International Journal of Hybrid Intelligent Systems*, 1(1):22–35, 2004.

[17] J. Dias, M. E. Captivo, and J. Clímaco. A memetic algorithm for multi-objective dynamic location problems. *Journal of Global Optimization*, 42(2):221–253, 2008.

[18] R. Li, T. R. Mersch, O. X. Wen, A. Kaylani, and G. C. Anagnostopoulos. Multi-objective memetic evolution of art-based classifiers. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, 2010.

[19] H. Kakuta and H. Mori. A multi-objective memetic algorithm for probabilistic transmission network expansion planning. In *IEEE International Conference on Systems Man and Cybernetics*, pages 1414–1419. IEEE, 2010.

[20] Y. F. Li, N. Pedroni, and E. Zio. A memetic evolutionary multi-objective optimization method for environmental power unit commitment. *IEEE Transactions on Power Systems*, 28(3):2660–2669, 2013.

[21] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[22] F. J. Rodriguez, C. Garcia-Martinez, and M. Lozano. Hybrid metaheuristics based on evolutionary algorithms and simulated annealing: Taxonomy, comparison, and synergy test. *IEEE Transactions on Evolutionary Computation*, 16(6):787–800, 2012.

[23] L. Han, Z. R. Jing, M. M. Wei, and L. X. Zhang. Adaptive extended kalman filter based on genetic algorithm for tightly-coupled integrated inertial and gps navigation. In *Second International Conference on Intelligent Computation Technology and Automation*, volume 1, pages 520–524. IEEE, 2009.

[24] D. J. Jwo and F. C. Chung. Fuzzy adaptive unscented kalman filter for ultra-tight gps/ins integration. In *International Symposium on Computational Intelligence and Design*, volume 2, pages 229–235. IEEE, 2010.

[25] Q. H. Nguyen, Y. S. Ong, and N. Krasnogor. A study on the design issues of memetic algorithm. In *IEEE Congress on Evolutionary Computation*, pages 2390–2397. IEEE, 2007.

[26] N. Krasnogor and J. Smith. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *IEEE Transactions on Evolutionary Computation*, 9(5):474–488, 2005.

[27] H. Ishibuchi, T. Yoshida, and T. Murata. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation*, 7(2):204–223, 2003.