

# COMPUTATIONAL COMPLEXITY REDUCTION TECHNIQUES FOR QUADRATURE KALMAN FILTERS

*Pau Closas, Jordi Vilà-Valls and Carles Fernández-Prades*

Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), 08860, Barcelona, Spain  
Statistical Inference for Communications and Positioning Department  
Email: {pclosas, jvila, cfernandez}@cttc.cat

## ABSTRACT

Nonlinear filtering is a major problem in statistical signal processing applications and numerous techniques have been proposed in the literature. Since the seminal work that led to the Kalman filter to the more advanced particle filters, the goal has been twofold: to design algorithms that can provide accurate filtering solutions in general systems and, importantly, to reduce their complexity. If Gaussianity can be assumed, the family of sigma-point KFs is a powerful tool that provide competitive results. It is known that the quadrature KF provides the best performance among the family, although its complexity grows exponentially on the state dimension. This article details the asymptotic complexity of the legacy method and discusses strategies to alleviate this cost, thus making quadrature-based filtering a real alternative in high-dimensional Gaussian problems.

**Index Terms**— Quadrature Kalman filters, sigma-point filters, complexity reduction, marginalization, sparse grids, high-dimensional.

## 1. INTRODUCTION

The Bayesian filtering problem involves the recursive estimation of time-varying unknown states of a system using the incoming flow of information along some prior statistical knowledge about the variations of such states. A discrete state-space model is expressed as

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}) + \mathbf{v}_k; \mathbf{y}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{n}_k, \quad (1)$$

where  $\mathbf{x}_k \in \mathbb{R}^{n_x}$  is the hidden state of the system at time  $k$ ,  $\mathbf{f}_{k-1}(\cdot)$  is a known, possibly nonlinear, function of the states; and  $\mathbf{v}_k$  is referred to as process noise;  $\mathbf{y}_k \in \mathbb{R}^{n_y}$  is the measurement at time  $k$ ,  $\mathbf{h}_k(\cdot)$  is a known, possibly nonlinear, function, which relates measurements with states; and  $\mathbf{n}_k$  is referred to as measurement noise, independent of  $\mathbf{v}_k$ . The optimal Bayesian filtering solution is given by the marginal distribution  $p(\mathbf{x}_k | \mathbf{y}_{1:k})$ , which gathers all the information about the system contained in the available observations, with  $\mathbf{y}_{1:k} = \{\mathbf{y}_1, \dots, \mathbf{y}_k\}$ . The Kalman filter (KF) provides the closed form solution to the optimal filtering problem in linear/Gaussian systems, assumptions that not always hold, reason why suboptimal techniques have to be used. A plethora of alternatives have been proposed to solve the nonlinear estimation problem, among them, the Extended KF (EKF) and the family of Sigma-Point KFs (SPKF) [1] within the Gaussian framework, and the family of

Sequential Monte Carlo (SMC) methods (a.k.a. Particle Filters (PF)) [2] for arbitrary noise distributions.

In the nonlinear Gaussian context, SPKF are usually preferred over standard SMC methods because of its simplicity and good performance. Considering similar computational complexity (i.e., number of sigma-points in the SPKF equal to the number of particles in the PF), SPKF have been shown to provide better performances under unimodal distributions [3, 4]. The main filters of the SPKF family are: the Unscented KF (UKF), the Cubature KF (CKF) and the Quadrature KF (QKF) [1]. Among them, the QKF is the one providing the best performances but the highest computational complexity. Such complexity is directly related to the number of deterministic sigma-points used in the filter. While the number of sigma-points grows linearly with the dimension of the state for the UKF and CKF, it grows exponentially for the QKF. This implies that the better performance comes at expenses of an exponential computational complexity increase. Therefore, one of the ultimate challenges within the SPKF family of filters is the design of efficient, low-complexity, QKF-based algorithms.

The main goal of the paper is first to analyze the computational complexity of the square-root version of the QKF, and then to detail three computational complexity reduction strategies for such filter. Notice that all of them target the reduction of the number of sigma-points without compromising the QKF performance. Namely, these approaches are *i*) partitioning of the state-space, *ii*) marginalization of linear substructures, and *iii*) generation of efficient quadrature sigma-points (sparse grids).

## 2. THE SQUARE-ROOT QUADRATURE KALMAN FILTER

Gaussian filters are of special interest by their broad applicability and their good balance between performance (i.e., estimation accuracy) and the required computational load. The optimal Bayesian recursive solution to the nonlinear filtering problem requires evaluating integrals that do not have analytical solution in the general case. A way to solve the problem is resorting to numerical methods.

Most numerical integration techniques consist of approximating the integrand by a polynomial in a region and then integrating the polynomial exactly. For instance, the Gauss-Hermite quadrature rule  $\{w_j, x_j\}_{j=1}^L$  is a form of Gaussian quadrature for approximating the value of integrals of type  $\int_{-\infty}^{+\infty} e^{-x^2} f(x) dx$ , such that

$$\int_{-\infty}^{+\infty} e^{-x^2} f(x) dx \approx \sum_{i=1}^L w_i f(x_i), \quad (2)$$

where  $L$  is the number of sample points used. The  $x_i$  are the roots of the physicists' version of the Hermite polynomial  $H_L(x)$  ( $i =$

This work has been partially funded by the European Commission in the Network of Excellence in Wireless COMMUNICATIONS NEWCOM# (contract n. 318306), the Spanish Ministry of Economy and Competitiveness through project TEC2012-39143 (SOSRAD) and by the Government of Catalonia (2014-SGR-1567).

---

**Algorithm 1** Generation of Sigma-Points and weights for the multidimensional Gauss-Hermite quadrature rule (full grid).

---

- 1: Set the number of points per dimension  $\alpha$ , and  $L = \alpha^{\dim(\mathbf{x})}$ .
  - 2: Set  $J_{i,i+1} = \sqrt{\frac{i}{2}}$ , where  $i = 1, \dots, (\alpha - 1)$ .
  - 3: Compute  $\lambda_i$ , the eigenvalues of  $\mathbf{J}$ .
  - 4: Set  $\xi_i = \sqrt{2\lambda_i}$ .
  - 5: Set  $\omega_i = (\nu_i)_1^2$ , where  $(\nu_i)_1$  is the first element of the  $i$ -th normalized eigenvector of  $\mathbf{J}$ .
  - 6: **if**  $\dim(\mathbf{x}) > 1$  **then**
  - 7:   Set  $\mathcal{S}_{1,:} = \xi_i$  and  $\varpi_{1,:} = \omega_i$
  - 8:   **for**  $m = 2$  to  $\dim(\mathbf{x})$  **do**
  - 9:     Set  $\mathcal{P}_{1:m-1,:} = \mathcal{S} \otimes \mathbf{1}_{1 \times \alpha}$
  - 10:     Set  $\mathcal{P}_{m,:} = \mathbf{1}_{1 \times \alpha} \otimes \mathcal{S}_{m-1,:}$
  - 11:     Set  $\Omega_{1:m-1,:} = \varpi \otimes \mathbf{1}_{1 \times \alpha}$
  - 12:     Set  $\Omega_{m,:} = \mathbf{1}_{1 \times \alpha} \otimes \varpi_{m-1,:}$
  - 13:     Set  $\mathcal{S} = \mathcal{P}$  and  $\varpi = \Omega$
  - 14:   **end for**
  - 15:    $\xi_i = \mathcal{P}_{:,i}$ , where  $i = 1, \dots, L$
  - 16:    $\omega_i = \prod_{l=1}^{\dim(\mathbf{x})} \Omega_{l,i}$
  - 17: **end if**
- 

1, 2, ..., L), defined as  $H_L(x) = (-1)^L e^{x^2} \frac{d^L}{dx^L} e^{-x^2}$ , and the associated weights  $w_i$  are given by  $w_i = \frac{2^{L-1} L! \sqrt{\pi}}{L^2 [H_{L-1}(x_i)]^2}$ .

An algorithm for generating the quadrature points and their corresponding weights for unidimensional problems was proposed in [5], and Algorithm 1 extends it to the multidimensional case, where  $\otimes$  denotes the tensor product. By construction, the number of sigma points growth exponentially with the dimension of the state-space  $\mathbf{x}_k$ . Assuming  $\alpha$  points per dimension, the Gauss-Hermite quadrature rule generates  $L = \alpha^{n_x}$  sigma points. This numerical integration method, when used in the SPKF framework, results in the quadrature Kalman filter (QKF) algorithm [6].

In the nonlinear filtering context, errors are likely to be introduced in the system from various sources (round-off errors, errors of nonlinear functional evaluations when they are computed in a system with limited arithmetic precision, etc). Those errors tend to accumulate over time, causing the filter to diverge. In order to improve the numerical accuracy and preservation of the covariance matrix symmetry (resulting in improved numerical stability), [7] proposed the *square-root* version of the algorithm, where the covariance matrix  $\Sigma_{\mathbf{x}}$  is decomposed into  $\Sigma_{\mathbf{x}} = \mathbf{S}_{\mathbf{x}} \mathbf{S}_{\mathbf{x}}^T$ , and it is  $\mathbf{S}_{\mathbf{x}}$  what is passed to the next time step. The resulting method, described in Algorithm 2, is used along this paper as a baseline for the complexity analysis.

### 3. ON THE SR-QKF COMPUTATIONAL COMPLEXITY

Real-time applications use to run in embedded digital processors with limited computational, storage and power consumption resources. For that reason, it is important to analyze an algorithm to determine the amount of resources (such as time and storage) necessary to execute it. In this Section, we will be concerned only with asymptotic time complexity,  $\mathcal{O}(\cdot)$ . The meaning of this notation is the following: a function  $p(z)$  is  $\mathcal{O}(g(z))$  if and only if there exist a real, positive constant  $K$  and a positive integer  $z_0$  such that  $p(z) \leq K g(z)$ ,  $\forall z \geq z_0$ . The time complexity of an algorithm can be viewed as the number of basic operations it performs.

Table 1 analyzes the time complexity of a single iteration of Algorithm 2. We assumed that the dimension of the state-space is higher than the observation state (so  $n_x > n_y$ ), and:

- noises covariances are constant and thus not updated, *i.e.*

---

**Algorithm 2** Square-root sigma-point Kalman filter (SR-SPKF).

---

**Require:**  $\mathbf{y}, \hat{\mathbf{x}}_0, \Sigma_{\mathbf{x},0} = \mathbf{S}_{\mathbf{x},0|0} \mathbf{S}_{\mathbf{x},0|0}^T, \Sigma_{\mathbf{v},k}, \Sigma_{\mathbf{n},k}$ .

- Initialization:*
- 1: Define sigma-points and weights  $\{\xi_i, \omega_i\}_{i=1,\dots,L}$  by using Algorithm 1, Algorithm 3 or any other rule.
  - 2: Set  $\mathbf{W} = \text{diag}(\sqrt{\omega_i})$
- Tracking:*
- 3: **for**  $k = 1$  to  $\infty$  **do**
  - Time update:**
  - 4: Evaluate the sigma points:
 
$$\mathbf{x}_{i,k-1|k-1} = \mathbf{S}_{\mathbf{x},k-1|k-1} \xi_i + \hat{\mathbf{x}}_{k-1|k-1}, \quad i = 1, \dots, L.$$
  - 5: Evaluate the propagated sigma points:
 
$$\tilde{\mathbf{x}}_{i,k|k-1} = \mathbf{f}_k(\mathbf{x}_{i,k-1|k-1}), \quad i = 1, \dots, L.$$
  - 6: Estimate the predicted state:
 
$$\hat{\mathbf{x}}_{k|k-1} = \sum_{i=1}^L \omega_i \tilde{\mathbf{x}}_{i,k|k-1}.$$
  - 7: Estimate the square-root of the predicted error covariance:
 
$$\mathbf{S}_{\mathbf{x},k|k-1} = \mathbf{Tria} \left( \left[ \tilde{\mathcal{X}}_{k|k-1} \mid \mathbf{S}_{\Sigma_{\mathbf{v},k}} \right] \right),$$
 where:  $\mathbf{S}_{\Sigma_{\mathbf{v},k}}$  is a square-root factor of  $\Sigma_{\mathbf{v},k}$  such that  $\Sigma_{\mathbf{v},k} = \mathbf{S}_{\Sigma_{\mathbf{v},k}} \mathbf{S}_{\Sigma_{\mathbf{v},k}}^T$ , and
 
$$\tilde{\mathcal{X}}_{k|k-1} = [\tilde{\mathbf{x}}_{1,k|k-1} - \hat{\mathbf{x}}_{k|k-1} \quad \dots \quad \tilde{\mathbf{x}}_{L,k|k-1} - \hat{\mathbf{x}}_{k|k-1}] \mathbf{W}.$$
  - Measurement update:**
  - 8: Evaluate the sigma points:
 
$$\mathbf{x}_{i,k|k-1} = \mathbf{S}_{\mathbf{x},k|k-1} \xi_i + \hat{\mathbf{x}}_{k|k-1}, \quad i = 1, \dots, L.$$
  - 9: Evaluate the propagated sigma points:
 
$$\tilde{\mathbf{y}}_{i,k|k-1} = \mathbf{h}_k(\mathbf{x}_{i,k|k-1}), \quad i = 1, \dots, L.$$
  - 10: Estimate the predicted measurement:
 
$$\hat{\mathbf{y}}_{k|k-1} = \sum_{i=1}^L \omega_i \tilde{\mathbf{y}}_{i,k|k-1}.$$
  - 11: Estimate the square-root factor of the innovation covariance matrix:
 
$$\mathbf{S}_{\mathbf{y},k|k-1} = \mathbf{Tria} \left( \left[ \tilde{\mathcal{Y}}_{k|k-1} \mid \mathbf{S}_{\Sigma_{\mathbf{n},k}} \right] \right),$$
 where:
 
$$\tilde{\mathcal{Y}}_{k|k-1} = [\tilde{\mathbf{y}}_{1,k|k-1} - \hat{\mathbf{y}}_{k|k-1} \quad \dots \quad \tilde{\mathbf{y}}_{L,k|k-1} - \hat{\mathbf{y}}_{k|k-1}] \mathbf{W}.$$
  - 12: Estimate the cross-covariance matrix
 
$$\Sigma_{\mathbf{x}\mathbf{y},k|k-1} = \mathcal{X}_{k|k-1} \mathcal{Y}_{k|k-1}^T,$$
 where:
 
$$\mathcal{X}_{k|k-1} = [\mathbf{x}_{1,k|k-1} - \hat{\mathbf{x}}_{k|k-1} \quad \dots \quad \mathbf{x}_{L,k|k-1} - \hat{\mathbf{x}}_{k|k-1}] \mathbf{W}.$$
  - 13: Estimate the Kalman gain
 
$$\mathbf{K} = \left( \Sigma_{\mathbf{x},k|k-1} / \mathbf{S}_{\mathbf{x},k|k-1}^T \right) / \mathbf{S}_{\mathbf{x},k|k-1}.$$
  - 14: Estimate the updated state
 
$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K} (\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}).$$
  - 15: Estimate the square-root factor of the corresponding error covariance:
 
$$\mathbf{S}_{\mathbf{x},k|k} = \mathbf{Tria} \left( \left[ \mathcal{X}_{k|k-1} - \mathbf{K} \mathcal{Y}_{k|k-1} \mid \mathbf{K} \mathbf{S}_{\Sigma_{\mathbf{n},k}} \right] \right).$$
  - 16: **end for**
- 

$$\Sigma_{\mathbf{v},k} = \Sigma_{\mathbf{v},0} \text{ and } \Sigma_{\mathbf{n},k} = \Sigma_{\mathbf{n},0},$$

- the multiplication of matrices of size  $n_1 \times n_2$  and  $n_2 \times n_3$  costs  $\mathcal{O}(n_1 n_2 n_3)$ , and
- the triangularization operator  $\mathbf{Tria}(\mathbf{A})$ , where  $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2}$ ,  $n_1 < n_2$ , is implemented with the thin QR decomposition [8, §5.2], which complexity is  $\mathcal{O}(n_2 n_1^2)$ .

No assumptions were made regarding the evaluation cost for functions  $\mathbf{f}(\cdot)$  and  $\mathbf{h}(\cdot)$ , although it is well-known that nonlinear functions tend to be computationally expensive, and often require look-up tables. We left those costs generic, denoting them as  $\mathcal{C}_f$  and  $\mathcal{C}_h$  respectively. Inspecting Table 1, it seems clear that parameter  $L = \alpha^{n_x}$  dominates the SR-QKF complexity and therefore, reducing the number of generated sigma-points has a direct impact in reducing the computational complexity of the overall filter.

Computation	Operation	Size	Cost
$\mathbf{S}_{\mathbf{x},k-1 k-1}\boldsymbol{\xi}_i + \hat{\mathbf{x}}_{k-1 k-1}$	$L$ matrix–vector products	$L \times n_x \times n_x$	$\mathcal{O}(Ln_x^2)$
$\mathbf{f}(\mathbf{x}_{i,k-1 k-1})$	$L$ evaluations of $\mathbf{f}(\cdot)$	$L$	$\mathcal{O}(L\mathcal{C}_f)$
$\sum_{i=1}^L \omega_i \tilde{\mathbf{x}}_{i,k k-1}$	scalar–vector product and sum	$L \times n_x$	$\mathcal{O}(Ln_x)$
$\text{Tria} \left( \begin{array}{c} \tilde{\mathcal{X}}_{k k-1} \\ \mathbf{S}_{\Sigma_{\mathbf{v},k}} \end{array} \right)$	Thin QR	$n_x \times (L + n_x)$	$\mathcal{O}((L + n_x)n_x^2)$
$\mathbf{S}_{\mathbf{x},k k-1}\boldsymbol{\xi}_i + \hat{\mathbf{x}}_{k k-1}$	$L$ matrix–vector products	$L \times n_x \times n_x$	$\mathcal{O}(Ln_x^2)$
$\mathbf{h}(\mathbf{x}_{i,k k-1})$	$L$ evaluations of $\mathbf{h}(\cdot)$	$L$	$\mathcal{O}(L\mathcal{C}_h)$
$\sum_{i=1}^L \omega_i \tilde{\mathbf{y}}_{i,k k-1}$	scalar–vector product and sum	$L \times n_y$	$\mathcal{O}(Ln_y)$
$\text{Tria} \left( \begin{array}{c} \mathcal{Y}_{k k-1} \\ \mathbf{S}_{\Sigma_{\mathbf{n},k}} \end{array} \right)$	Thin QR	$n_y \times (L + n_y)$	$\mathcal{O}((L + n_y)n_y^2)$
$\mathcal{X}_{k k-1}\mathcal{Y}_{k k-1}^T$	matrix–matrix product	$n_x \times L \times n_y$	$\mathcal{O}(Ln_x n_y)$
$(\Sigma_{\mathbf{x},k k-1}/\mathbf{S}_{\mathbf{x},k k-1}^T) / \mathbf{S}_{\mathbf{x},k k-1}$	2 backward substitutions	$n_x \times n_x$	$\mathcal{O}(2n_x^2)$
$\hat{\mathbf{x}}_{k k-1} + \mathbf{K}(\mathbf{y}_k - \hat{\mathbf{y}}_{k k-1})$	matrix–vector product	$n_x \times n_y$	$\mathcal{O}(n_x n_y)$
$\text{Tria} \left( \begin{array}{c} \mathcal{X}_{k k-1} - \mathbf{K}\mathcal{Y}_{k k-1} \\ \mathbf{K}\mathbf{S}_{\Sigma_{\mathbf{n},k}} \end{array} \right)$	Thin QR	$n_x \times (L + n_x)$	$\mathcal{O}((L + n_x)n_x^2)$
	matrix–matrix product $\mathbf{K}\mathcal{Y}_{k k-1}$	$n_x \times n_y \times L$	$\mathcal{O}(Ln_x n_y)$
	matrix–matrix product $\mathbf{K}\mathbf{S}_{\Sigma_{\mathbf{n},k}}$	$n_x \times n_y \times n_y$	$\mathcal{O}(n_x n_y^2)$

**Table 1:** Asymptotic time complexity analysis of the main operations involved in the square-root sigma-point Kalman filter (Algorithm 2).

#### 4. COMPLEXITY REDUCTION FOR QUADRATURE-BASED FILTERING

This section presents state-of-the-art approaches for the reduction of the inherent complexity in quadrature-based filters. The Multiple QKF and the Marginalized QKF approaches reduce the dimensionality of the problem by exploiting possible substructures in the state-space, while the sparse QKF reduces the total number of points of the quadrature rule,  $L$ , while maintaining accuracy.

##### 4.1. Multiple QKF

The Multiple QKF approach [9] exploits the problem substructures (specifically, states with low or null cross-correlation) to reduce the number of required sigma points. The state space is partitioned into  $S$  subspaces, and the complexity reduction is based on grouping the states which are highly correlated and splitting the uncorrelated ones (or with low correlation) into different subspaces. In such a way, a bank of QKFs is in charge of filtering the resulting, low-dimensional,  $S$  subspaces thus reducing the number of sigma-points per filter. The different filters are indeed interacting since, in general, measurements are typically dependent on the entire state vector and thus some sort of information exchanges has to take place.

Whereas the number of points used by a QKF is  $L = \alpha^{n_x}$ , the total number of points generated by the Multiple QKF is  $L_M = \sum_{s=1}^S \alpha^{n_x^{(s)}}$ , with  $n_x^{(s)}$  being the dimension of each subspace. It was proved that  $L_M$  is lower than  $L$  if  $\alpha > S^{\frac{1}{S-1}}$  for  $S \geq 2$  and  $\alpha > 2$  [9]. These conditions are general accomplished since the number of points per dimension  $\alpha$  is typically set to 3 or 5 in practice, and at least two partitions are done in the Multiple QKF and thus  $S \geq 2$ . This result proves that the MQKF is always reducing the number of generated quadrature points, and thus the computational complexity of the overall filter. Accuracy degradation was shown to be moderated in problems with low cross-correlation between states.

##### 4.2. Marginalized QKF

Another way of alleviating the dimensionality problem and improving the overall filter performance is to marginalize linear states, that

is, linear substructures in the general nonlinear dynamic system. The basic idea is that a Kalman filter (KF) can optimally deal with these states, while reducing the dimension of the state-space that the sigma-point Kalman filter (SPKF) has to explore. The procedure was proposed in [10] in the context of particle filtering. The algorithm was termed marginalized particle filter (MPF), although the same concept is also referred to as Rao-Blackwellized PF (RBPf) in other works [11, 2]. The latter nomenclature is because marginalization resorts to a general result referred to as the Rao-Blackwell theorem [12], which shows that the performance of an estimator can be improved by using information about conditional probabilities.

The so-called Marginalized SPKF is able to reduce the computational complexity of legacy SPKF techniques by a factor that depends on the dimension of the linear subspace. The Marginalized SPKF has key differences to the MPF, mainly due to the way the numerical integrals are approximated through deterministic and random points, respectively.

In the MPF, the nonlinear subspace is estimated using a PF where each particle has a dedicated KF to cope with the linear states. Indeed, linear and nonlinear subspaces are conditionally dependent and its interaction should be handled with special care. The way the MPF propagates the nonlinear state estimates uncertainty to the linear subspace is via the particle approximation, which characterizes the posterior distribution. This can be done because each particle follows a trajectory throughout the whole PF operation, thus each KF operates in accordance with a given trajectory, that is, always associated to the same particle. The resampling stage of the PF does not cause any problem because the KFs are replicated or deleted together with the corresponding particles.

In contrast, for the Marginalized SPKF, the key point is to notice that the sigma-points are reset in each SPKF prediction and update step, therefore the concept of trajectory of a given sigma-point does not exist. In this context, considering a KF for each deterministic sigma-point is not theoretically funded and has no practical meaning. This fact directly leads to the general Marginalized SPKF architecture, where a single SPKF interacts with a single KF, as opposite to the multiple KFs used in the MPF.

---

**Algorithm 3** Generation of Sigma-Points and weights for a multi-dimensional sparse quadrature rule (sparse grid).

---

```

1: Set  $n_x = \dim\{\mathbf{x}\}$  and the accuracy level  $\mathcal{L}$ .
2: for  $q = \mathcal{L} - n_x$  to  $\mathcal{L} - 1$  do
3:   Determine  $\mathbb{N}_q^{n_x}$  as defined in (3).
4:   for each element  $\mathbf{i}$  in  $\mathbb{N}_q^{n_x}$  do
5:     Compute  $\xi_{i_1}, \dots, \xi_{i_{n_x}}$  and  $\omega_{i_1}, \dots, \omega_{i_{n_x}}$  as in steps 2-5 of
     Algorithm 1 with  $L = \alpha = i_j$  (or with any other
     suitable quadrature rule, such as Kronrod–Patterson’s).
6:     Compute  $\xi_{i_1} \otimes \xi_{i_2} \otimes \dots \otimes \xi_{i_{n_x}}$ 
7:     for each point in  $\xi_{i_1} \otimes \xi_{i_2} \otimes \dots \otimes \xi_{i_{n_x}}$  do
8:       if the point is new then
9:         Add it to the list of sigma points.
10:        Assign a new index  $s$  to the point.
11:        Calculate its weight as:

$$\omega_s = (-1)^{\mathcal{L}-1-q} C_{n_x-1}^{\mathcal{L}-1-q} \prod_{k=1}^{n_x} \omega_{i_k}$$

12:       else
13:         Update the old weight by:

$$\omega_s = \omega_s + (-1)^{\mathcal{L}-1-q} C_{n_x-1}^{\mathcal{L}-1-q} \prod_{k=1}^{n_x} \omega_{i_k}$$

14:       end if
15:     end for
16:   end for
17: end for

```

---

### 4.3. Sparse QKF

The sparse grid method is a general numerical discretization technique for multidimensional integrals. This approach constructs a multidimensional quadrature rule by a special truncation (introduced by Smolyak) of the tensor product expansion of a one-dimensional quadrature rule, with a notable reduction in the total number of points while maintaining accuracy. The Smolyak rule with accuracy level  $\mathcal{L}$  is the one that integrates complete polynomials of total order  $2\mathcal{L} - 1$  exactly.

Defining  $C_{n_x-1}^{\mathcal{L}-1-q} = \binom{n_x-1}{\mathcal{L}-1-q}$  as the binomial coefficient, the set of accuracy level sequences  $\mathbb{N}_q^{n_x}$  as

$$\mathbb{N}_q^{n_x} = \begin{cases} \mathbb{N}_q^{n_x} = \{\mathbf{i} \in \mathbb{N}^{n_x} : \sum_{d=1}^{n_x} i_d = n_x + q\} & q \geq 0 \\ \mathbb{N}_q^{n_x} = \{\emptyset\} & q < 0 \end{cases}$$

for any integer  $q$  in the range  $\mathcal{L} - n_x \leq q \leq \mathcal{L} - 1$  (for example,  $\mathbb{N}_2^2 = \{[1, 3], [2, 2], [3, 1]\}$ ), the sparse quadrature rule [13] can be generated as shown in Algorithm 3.

Other suitable quadrature rules than the Gauss-Hermite can be used in step 5 of Algorithm 3. A sparse grid is constructed by summing a set of product rules. A feature of Gauss-Hermite rules is that they are not *nested* (i.e., a few of the points will be repeated across several sparse grids). The Kronrod-Patterson rule extends the unidimensional quadrature point set  $\xi_{\ell-1}$  by adding a new set of symmetric points, thus achieving that the quadrature set of each accuracy level is a subset of the set of its successors. This feature implies a further reduction in the total number of points. The sparse Kronrod-Patterson quadrature Kalman filter was proposed in [14], showing slightly better accuracy than the sparse Gauss-Hermite Quadrature rule for the same number of points. By using Algorithm 3 in step 1 of Algorithm 2, one obtains the square-root version, with improved numerical stability.

The number of points in sparse rules are bounded to scale polynomially with  $n_x$ , as  $\mathcal{O}(\alpha(\log \alpha)^{n_x-1})$ . For instance, the Gauss-Hermite rule for  $\mathcal{L} = 3$  uses  $L = 2n_x^2 + 6n_x + 1$  points (see [14])

and, according to Table 1, the complexity of the sparse SR-QKF is  $\mathcal{O}(n_x^4)$ . This is a substantial saving in complexity with respect to the SR-QKF with  $\alpha = 3$ , which is  $\mathcal{O}(n_x^2 3^{n_x})$ , with no penalty in numerical accuracy.

## 5. CONCLUSIONS

The QKF is a powerful technique to deal with nonlinear/Gaussian filtering problems with competitive results compared to PF solutions. However, its complexity grows exponentially with the dimension of the problem. This paper detailed the asymptotic complexity of the method and discussed various techniques aimed at reducing the complexity of the QKF. More precisely, techniques based on exploiting particular substructures in the state-space model and sparse grid versions of the quadrature rules were mentioned. It is worth noting that these approaches are complementary and one could conceive algorithms incorporating those enhancements simultaneously, thus enabling the use of QKF in large dimensional systems.

## 6. REFERENCES

- [1] P. Stano et al., “Parametric Bayesian filters for nonlinear stochastic dynamical systems: a survey,” *IEEE Trans. on Cybernetics*, vol. 43, no. 6, pp. 1607–1624, Dec. 2013.
- [2] A. Doucet, N. De Freitas, and N. Gordon, Eds., *Sequential Monte Carlo methods in practice*, Springer, 2001.
- [3] C. Fernández-Prades and J. Vilà-Valls, “Bayesian nonlinear filtering using quadrature and cubature rules applied to sensor data fusion for positioning,” in *Proc. of the IEEE Intl. Comm. Conf. (ICC’10)*, Cape Town, South Africa, May 2010.
- [4] F. Gustafsson, “Particle filter theory and practice with positioning applications,” *Aerospace and Electronic Systems Magazine, IEEE*, vol. 25, no. 7, pp. 53–82, July 2010.
- [5] G.H. Golub and J.H. Welsch, “Calculation of Gauss quadrature rules,” *Mathematics of Computation*, vol. 23, no. 106, pp. 221–230, April 1969.
- [6] K. Ito and K. Xiong, “Gaussian filters for nonlinear filtering problems,” *IEEE Trans. Automat. Contr.*, vol. 45, no. 5, pp. 910–927, May 2000.
- [7] I. Arasaratnam and S. Haykin, “Square-root quadrature Kalman filtering,” *IEEE Trans. Signal Processing*, vol. 56, no. 6, pp. 2589–2593, June 2008.
- [8] G. H. Golub and C. F. van Loan, *Matrix Computations*, The John Hopkins University Press, 3rd edition, 1996.
- [9] P. Closas, C. Fernández-Prades, and J. Vilà-Valls, “Multiple quadrature Kalman filtering,” *IEEE Trans. on Sig. Proc.*, vol. 60, no. 12, pp. 6125–6137, December 2012.
- [10] T. Schon, F. Gustafsson, and P-J. Nordlund, “Marginalized particle filters for mixed linear/nonlinear state-space models,” *IEEE Trans. on Sig. Proc.*, vol. 53, no. 7, pp. 2279–2289, 2005.
- [11] R. Chen and J. S. Liu, “Mixture Kalman filters,” *Journal of the Royal Stat. Soc. B*, vol. 62, pp. 493–508, 2000.
- [12] G. Casella and R. Berger, *Statistical Inference*, Duxbury Resource Center, June 2001.
- [13] B. Jia, M. Xin, and Y. Cheng, “Sparse Gauss-Hermite quadrature filter with application to spacecraft attitude estimation,” *J. of Guid., Cont., and Dyn.*, vol. 34, no. 2, pp. 367–379, 2011.
- [14] H. Chen, X. Cheng, C. Dai, and C. Ran, “Accuracy, efficiency and stability analysis of sparse-grid quadrature Kalman filter in near space hypersonic vehicles,” in *Proc. of IEEE/ION Pos., Loc. and Nav. Symp.*, Monterey, CA, May 2014, pp. 27–36.