AAS 15-423

GENERALIZED GAUSSIAN CUBATURE FOR NONLINEAR FILTERING

Richard Linares*
Los Alamos National Laboratory, Los Alamos, NM, 87544
John L. Crassidis†
University at Buffalo, State University of New York, Amherst, NY, 14260-4400

A novel method for nonlinear filtering based on a generalized Gaussian cubature approach is shown. Specifically, a new point-based nonlinear filter is developed which is not based on one-dimensional quadrature rules, but rather uses multi-dimensional cubature rules for Gaussian distributions. The new generalized Gaussian cubature filter is not in general limited to odd-order degrees of accuracy, and provides a wider range of order of accuracy. The method requires the solution of a set of nonlinear equations for finding optimal cubature points, but these equations are only required to be solved once for each state dimensional and order of accuracy. This rule is also extended to anisotropic cases where the order of accuracy is not isotropic in dimension. This method allows for tuning of the cubature rules to develop problem specific rules that are optimal for the given problem. The generalized Gaussian cubature filter is applied to benchmark problems in astrodynamics, and it is compared against existing nonlinear filtering methods.

INTRODUCTION

Sequential state estimation has been successfully applied to number of problems in many disciplines including; aircraft tracking, satellite orbit determination,[1] spacecraft attitude determine,[2,3] and many others.[4,5,6,7] The main goal of sequential state estimation is to use measurements of the system of interest along with dynamic model to determine an estimate of current state and some measure of error in this estimate. The Bayesian framework provides a mechanism to completely determine the probability space of the system state but the solution is not in general solvable for nonlinear systems. Several approach

Optimal integration approximation theory in one dimension is well-known, and many rules have been developed for different weighting functions and integration domains. In this work two distinctions are made; one between rules in one dimension and rules in multiple dimensions, and another between whether rules that are based on classical types of theory or more generalized forms. Integration rules in one dimension are referred to as "quadrature rules" and integration rules in multi dimensions are referred to as "cubature rules." Likewise integration rules are referred to as classical or generalized defined by their individual based theory.

*Director's Postdoctoral Fellow, Intelligence and Space Research. Email: linares@lanl.gov.
†CUBRC Professor in Space Situational Awareness, Department of Mechanical & Aerospace Engineering. Email: johnc@buffalo.edu.

The Gaussian quadratures rules in particular involve the optimal integration of polynomials with the minimum number of nodes. It is well-known how to construct Gaussian quadratures for one-dimensional integrals. Rules based on orthogonal polynomials are also well-known, and are referred to as Classical Gaussian Quadrature (CGQ) in this work. Recently a different class of quadrature rules has been developed that is not based on orthogonal polynomials, but rather on the integration of general classes of functions. These rules are referred to as Generalize Gaussian Quadrature (GGQ) in this work.[8]

For higher-dimensional problems optimal integration rules are very important. Integrals for dimensions greater than one still remain a challenge. The most straightforward way to construct these rules with tensor products of one-dimensional Classical Gaussian Cubature (CGC) rules.[9, 10, 11] These tensor products produce rules that are far from optimal, and require much more nodes than needed to achieve a pre-selected desirable precision. In particular the number of nodes increases rapidly with dimension, making these rules un-usable for dimensions greater than 20. This current work investigates a technique called generalized Gaussian cubature where the rules are constructed to integrate a set of function exactly. Also, this work develops an approach to form generalized Gaussian cubature rules for Gaussian mixture distributions.

PROBLEM STATEMENT

For Gaussian nonlinear filters the key computational challenge is evaluating Gaussian integrals of the following form:

$$\mathcal{I}(g) = \int_{\mathcal{R}^d} g(\mathbf{x}) \mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu}, P\right) d\mathbf{x} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^d$, $g : \mathbb{R}^d \to \mathbb{R}^1$, and $\mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu}, P\right)$ denotes a multivariate normal probability density function for $\mathbf{x}$ with mean and covariance $\boldsymbol{\mu}$ and P, respectively. The sigma-point filtering and smoothing methods in general can be viewed as methods which approximate the integral in Eq. (1) as

$$\int_{\mathcal{R}^d} g(\mathbf{x}) \mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu}, P\right) d\mathbf{x} \approx \sum_{i=1}^{m} W_i g(\mathbf{x}_i) \tag{2}$$

where W_i and $\mathbf{x}_i$ are predefined weights and sigma points, respectively. The Gaussian integral has a solution in general using a change of variables, called stochastic decoupling, where sigma-points are developed to solve the following integral:

$$\int_{\mathcal{R}^d} \tilde{g}(\boldsymbol{\eta}) \mathcal{N}\left(\boldsymbol{\eta}|0, I\right) d\boldsymbol{\eta} \approx \sum_{i=1}^{m} W_i \tilde{g}(\boldsymbol{\eta}_i) \tag{3}$$

where $\tilde{g} = g(\boldsymbol{\mu} + \sqrt{P}\boldsymbol{\eta})$ and $P = \sqrt{P}\sqrt{P}^T$. Sigma-point methods differ in the selection of sigma-points, $\boldsymbol{\eta}$, and weights, W_i, to solve the integral in Eq. (1). In general the function $\tilde{g}$ can take on any form. Therefore sigma-point methods are designed with the goal to be exact, and/or accurate for a class of functions with the hope that $\tilde{g}$ is well approximated by this class.

Typically the class of functions considered are polynomial functions of several variables. For the purpose of this work the order of the multi-dimensional polynomials will be defined by its the monomial with the largest multi-index L^1 norm. The multi-index is denoted by $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$, where each α_i is the order of i^{th} variable in a given monomial. Then for a given $\boldsymbol{\alpha}$ the set of variables $\mathbf{x}$ monomial in variables $x_1, \ldots, x_n$ of index $\boldsymbol{\alpha}$ is given by

$$\mathbf{x}^{\boldsymbol{\alpha}} = (x_1^{\alpha_1} \ldots x_d^{\alpha_d}) \tag{4}$$

The number $|\boldsymbol{\alpha}| = \sum_{i=1}^{d} \alpha_i$ is then called the degree of $\mathbf{x}^{\boldsymbol{\alpha}}$ and the order of a polynomial in multiple dimensions is determined by its highest degree monomial. The number of polynomial functions in d variables of degree less than ℓ is given by

$$N_\ell = \frac{(\ell + d)!}{\ell! d!} \tag{5}$$

The number of polynomials of degree greater than ℓ grows rapidly with dimension. This rapid growth is referred to the "curse of dimensionality." In the following sections nonlinear Bayesian filters based on Gaussian assumptions will be discussed. Tensor product and sparse gird tensor product cubature approximations to Eq. (1) will be shown. This will followed by a new generalized Gaussian cubature approach.

NONLINEAR GAUSSIAN FILTERING

In the general nonlinear filtering problem the goal is to determine an estimate of the state vector $\mathbf{x}$ from observations $\mathbf{y}$, where the following dynamic and observation models are given:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{w}_k \tag{6a}$$

$$\mathbf{y}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \tag{6b}$$

where $\mathbf{w}_k$ and $\mathbf{v}_k$ are zero-mean Gaussian noise processes with covariances $Q_k = E\{\mathbf{w}_k \mathbf{w}_k^T\}$ and $R_k = E\{\mathbf{v}_k \mathbf{v}_k^T\}$, respectively. Sequential Bayesian state estimation uses the a prior $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ (state uncertainty based on previous measurements) and the likelihood $p(\mathbf{y}_k|\mathbf{x}_k)$ (uncertainty based on measurements) to calculate the posterior probability density function (pdf) of the state conditioned on the current measurements through

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k) p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) d\mathbf{x}_k} \tag{7}$$

In between measurements the conditional pdf $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ is found using the Chapman-Kolmogorov equation:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}) d\mathbf{x}_{k-1} \tag{8}$$

The nonlinear filtering problem given in Eq. (6) can be solved with Eq. (7) and Eq. (8), but in general this problem is difficult to solve. Therefore, approximate Gaussian-based filters are often used. In the Gaussian filtering approach the likelihood and transition pdfs are assumed to be $p(\mathbf{x}_k|\mathbf{x}_{k-1}) = \mathcal{N}\left(\mathbf{x}_k - \mathbf{f}\left(\mathbf{x}_{k-1}\right);\, \mathbf{0},\, Q_k\right)$ and $p(\mathbf{y}_k|\mathbf{x}_k) = \mathcal{N}\left(\mathbf{y}_k - \mathbf{h}\left(\mathbf{x}_k\right);\, \mathbf{0},\, R_k\right)$, respectively. Then the nonlinear filter problem becomes estimating pre-update and post-update means and covariances ($\{\hat{\mathbf{x}}_{k|k-1}, P_{k|k-1}\}$ and $\{\hat{\mathbf{x}}_{k|k}, P_{k|k}\}$, respectively). The pre-update mean and covariance is found via propagating the state and covariance from the previous time-step to the current time-step, which is given by

$$\hat{\mathbf{x}}_{k|k-1} = \int_{\mathcal{R}^d} \mathbf{f}(\mathbf{x}_{k-1}) \mathcal{N}\left(\mathbf{x}_{k-1};\, \hat{\mathbf{x}}_{k-1|k-1},\, P_{k-1|k-1}\right) d\mathbf{x}_{k-1} \tag{9a}$$

$$P_{k|k-1} = \int_{\mathcal{R}^d} \mathbf{f}(\mathbf{x}_{k-1}) \mathbf{f}(\mathbf{x}_{k-1})^T \mathcal{N}\left(\mathbf{x}_{k-1};\, \hat{\mathbf{x}}_{k-1|k-1},\, P_{k-1|k-1}\right) d\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k|k-1} \hat{\mathbf{x}}_{k|k-1}^T + Q_k \tag{9b}$$

Then given measurements, the post-update mean and covariance equations are given by

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + L_k \left[\mathbf{y}_k - \mathbf{h}_k(\mathbf{x}_{k|k-1})\right] \tag{10a}$$

$$P_{k|k} = P_{k|k-1} - L_k P_{xy}^T \tag{10b}$$

where these equations follow the Gaussian filtering assumptions, and therefore have the Kalman filter structure. The terms in Eq. (10) are calculated as follows

$$L_k = P_{xy} \left(R_k + P_{yy}\right)^{-1} \tag{11a}$$

$$\hat{\mathbf{y}}_k = \int_{\mathcal{R}^d} \mathbf{h}\left(\mathbf{x}_k\right) \mathcal{N}\left(\mathbf{x}_k;\, \hat{\mathbf{x}}_{k|k-1},\, P_{k|k-1}\right) d\mathbf{x}_k \tag{11b}$$

$$P_{xy} = \int_{\mathcal{R}^d} \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}\right) \left(\mathbf{h}(\mathbf{x}_k) - \hat{\mathbf{y}}_k\right)^T \mathcal{N}\left(\mathbf{x}_k;\, \hat{\mathbf{x}}_{k|k-1},\, P_{k|k-1}\right) d\mathbf{x}_k \tag{11c}$$

$$P_{xx} = \int_{\mathcal{R}^d} \left(\mathbf{h}(\mathbf{x}_k) - \hat{\mathbf{y}}_k\right) \left(\mathbf{h}(\mathbf{x}_k) - \hat{\mathbf{y}}_k\right)^T \mathcal{N}\left(\mathbf{x}_k;\, \hat{\mathbf{x}}_{k|k-1},\, P_{k|k-1}\right) d\mathbf{x}_k \tag{11d}$$

Finally, all the integrals in Eqs. (9), (10), and (11) can be determined by solving the integral in Eq. (3). This paper proposes an improved solution to nonlinear filtering by improving the solution of Eq. (3) with the method discussed in the next section.

Tensor Product Cubature

The most straightforward way of developing cubature rules is using tensor products of one-dimensional Gaussian quadrature rules. Tensor Product Cubature (TPC) rules can be derived for Eq. (1) by noting that $\mathcal{N}\left(\boldsymbol{\eta};\mathbf{0},\, I\right) = \mathcal{N}\left(\eta_1;\, 0,\, 1\right) \cdots \mathcal{N}\left(\eta_d|;\, 0\,, 1\right)$, and the integral can be written as

$$\begin{aligned}\int_{\mathcal{R}^d} \tilde{g}(\boldsymbol{\eta}) \mathcal{N}\left(\boldsymbol{\eta};\, \mathbf{0},\, I\right) d\boldsymbol{\eta} = \\ \int_{\mathcal{R}} \mathcal{N}\left(\eta_1;\, 0,\, 1\right) \cdots \int_{\mathcal{R}} \mathcal{N}\left(\eta_2;\, 0,\, 1\right) \int_{\mathcal{R}} \tilde{g}(\boldsymbol{\eta}) \mathcal{N}\left(\eta_1;\, 0,\, 1\right) d\eta_1\, d\eta_2 \cdots d\eta_d\end{aligned} \tag{12}$$

where the integral above allows for a telescoping evaluation for the integral, and one-dimensional quadratures rules can be used for each of the η_i integrals. Then the multidimensional integral in Eq. (1) can be approximated using the tensor products of one-dimensional CGQ rules.[3] Each integral in Eq. (12) can be written as

$$\int_{\mathcal{R}^1} g(\eta_j) \mathcal{N}\left(\eta_j;\, 0,\, 1\right) d\eta_j = \sum_{i=1}^{m} g(\eta_j^i) W_j^i \tag{13}$$

where the one-dimensional nodes η_j^i and weights W_j^i are from one of the several known one-dimensional rules. Also, m denotes the number of points for the chosen level of accuracy. These rules include Clenshaw-Curtis, Gauss-Patterson, Gauss-Hermite, Gauss-Legendre, Gauss-Laguerre, generalized Gauss-Laguerre, and Gauss-Jacobi. Each of these rules is a CGQ for a given weight function contained in the integral and the domain of the integral. In other words each rule is optimal for a given form of the integral in Eq. (13).

Using the rule in Eq. (13), multidimensional rules are constructed with tensor products of the one-dimensional rules. Each one-dimensional rule is exact for polynomials of degree less than or

equal to $2m_j - 1$, where m_j is the number of points used for the rule in the j^{th} dimension. Different rules may be used along each dimension and therefore the index $\boldsymbol{\alpha}$. The multi-index is then given by $\boldsymbol{\alpha} = (\alpha_1, \cdots \alpha_n)$, which is used to define the accuracy level along each dimension. The tensor product cubature rule is then defined by

$$\mathcal{I}(g) \approx \sum_{i_1=1}^{\alpha_1} \cdots \sum_{i_n=1}^{\alpha_n} g(\eta_j^i,)(w_j^{i_1} \otimes \cdots \otimes w_j^{i_n}) \tag{14}$$

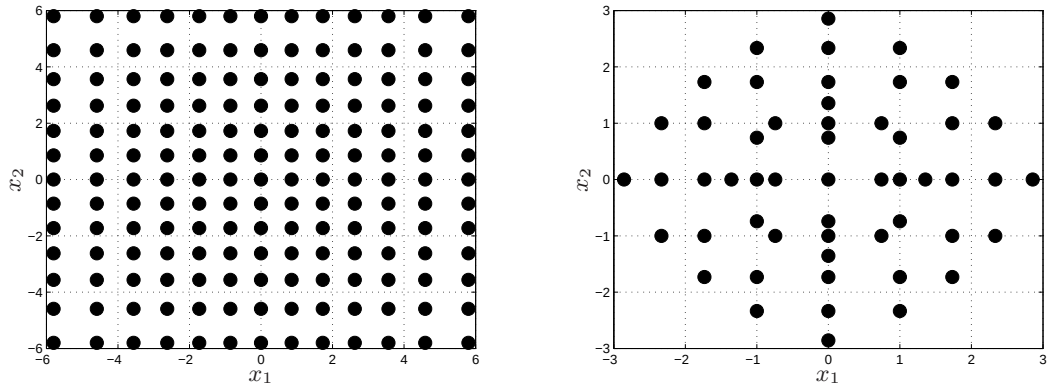
The total number of cubature points required for the rule in Eq. (14) is given by the product of each one-dimensional quadrature rule:

$$N_{\text{TPC}} = \prod_{i=1}^{d} \boldsymbol{\alpha}(i) \tag{15}$$

When the same level of accuracy is used for each dimension, the number of points for the tensor product quadrature rule is given by

$$N_{\text{TPC}} = m^d \tag{16}$$

From Eqs (15) and (16) it is clear that the number of points grows rapidly for both increasing level of accuracy m and dimension d. This growth in numerical complexity of the tensor product rule is related to the curse of dimensionality. The tensor product rule uses more points than is requires to achieve a given accuracy level. In general, the tensor product rule is only feasible for low-dimensional problems.



(a) Tensor Product Rule

(b) Smolyak Sparse Rule

Figure 1. Cubature Rules

Smolyak Sparse Grids

There are a number of applications that require the evaluation of the multidimensional integral in Eq. (1) for large number of integration variables. The number of nodes in the tensor product rule grows rapidly with accuracy level and dimension, but the growth rate is higher than required for achieving a given accuracy level, as previously mentioned. The sparse tensor product method

is a popular cubature method which is more efficient than tensor product-based rules. One sparse tensor product method is the Sparse Grid Cubature (SGC) method, which was first introduced by Smolyak[10] and further developed by others. These methods reduce the number of cubature points drastically while maintaining the desired accuracy level.

The Smolyak isotropic formulas will now be described, following from a description in Ref. 10. The Smolyak isotropic formula operator is denoted by $Q_\ell^{(N)}$, where ℓ is a level that is independent of dimension N. The Smolyak formulas reduce the number of nodes by only using tensor products with relatively small number of points. The Smolyak formulas are linear combinations of the product formulas that limit the growth of the number of nodes by limiting tensor products which result in large number of points. Another key property is in the definition of difference formulas:

$$\Delta^i = U_i - U_{i-1} \tag{17}$$

with $U_0 = 0$. For $i \geq 1$ the quadrature U_i can be written as

$$U_i = \sum_{i=1}^{l} \Delta^i \tag{18}$$

The ℓ^{th} degree quadrature can be written as the sum of difference formulas. The use of the difference formulas simplifies the sparse cubature formulas greatly for multidimensional cases. The isotropic Smolyak quadrature formula is given by

$$Q_\ell^{(N)} = \sum_{|\boldsymbol{\alpha}| \leq \ell + n} (\Delta^{\alpha_1} \otimes \cdots \otimes \Delta^{\alpha_n}) \tag{19}$$

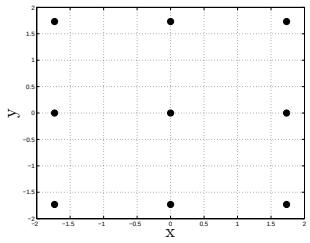
Note that $|\boldsymbol{\alpha}| = \alpha_1 + \cdots + \alpha_n$. Equivalently, the formula in Eq. (19) can be written as

$$Q_\ell^{(N)} = \sum_{\ell+1 \leq |\boldsymbol{\alpha}| \leq \ell+N} (-1)^{w+d-|\boldsymbol{\alpha}|} \begin{pmatrix} d-1 \\ w+d-|\boldsymbol{\alpha}| \end{pmatrix} (U^{\alpha_1} \otimes \cdots \otimes U^{\alpha_n}) \tag{20}$$

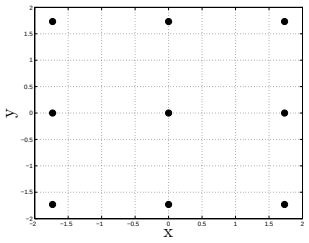
The form of Eq. (20) is also valid for the tensor product cubature rule in Eq. (12). This rule can be written by

$$Q_\ell^{(N)} = \sum_{\max |\boldsymbol{\alpha}| \leq \ell} (U^{\alpha_1} \otimes \cdots \otimes U^{\alpha_n}), \qquad \max |\boldsymbol{\alpha}| \equiv \max\{\ell_1, \ldots, \ell_N\} \tag{21}$$

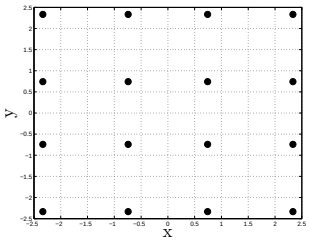
The difference between the full tensor product rules and the sparse tensor product rules is the index $\boldsymbol{\alpha}$ used in the summation. The full tensor rules sum over a hypercube of a multi-index space, and consider much more terms than are necessary for the given order accuracy, whereas the sparse rules sum over a simplex of terms less than or equal to the desired order. The simplex used for the sparse rules is defined by $|\boldsymbol{\alpha}| \leq \ell$. An example of a tensor rule and sparse rule is shown in Figure 1. Here one can see that the tensor rule uses far more points than necessary, whereas the sparse rule tends to focus the points more in areas of importance. Sparse rules require nesting of one-dimensional rules, and are limited by the properties of these one-dimensional rules. For example, one-dimensional rules can achieve an order of accuracy that is $2m - 1$, which means that only odd-order orders of accuracy can be achieved. The work in this papers derives a general approach for developing cubature rules that reduces the number of nodes used, thereby controlling the growth of nodes with increasing dimension. An adaptive multi-index space can be used to optimize the rule for the problem to be solved.



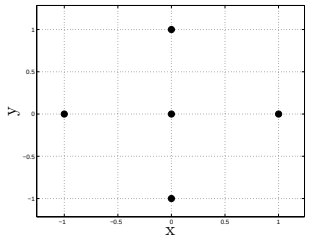
(a) Tensor GC Order 3



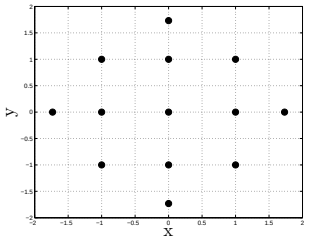
(b) Tensor GC Order 5



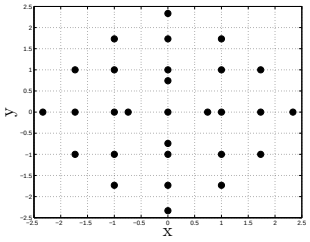
(c) Tensor GC Order 7



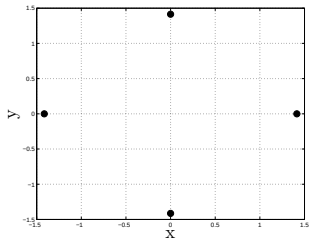
(d) Sparse GC Order 3



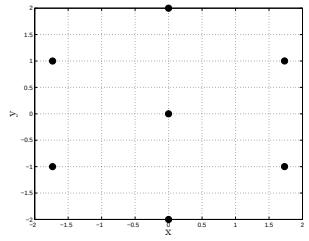
(e) Sparse GC Order 5



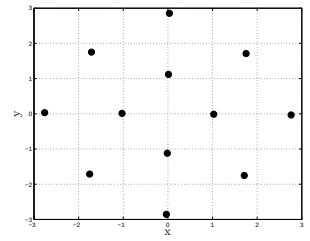
(f) Sparse GC Order 7



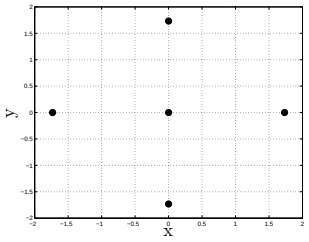
(g) GGC Order 3



(h) GGC Order 5



(i) GGC Order 7



(j) UKF

Figure 2. Cubature Points vs Order in 2 Dimensions

GENERALIZED GAUSSIAN CUBATURE

The extension of the classical theory of orthogonal polynomials to multiple dimensions is far from complete. By far the clearest approach seems to be extending GGQ to multidimensional functions. Throughout this work this extension will be called Generalize Gaussian Cubature (GGC). The GGC for multidimensional cases is designed to evaluate the following multidimensional integral:

$$\int_{\Omega} f(\boldsymbol{x}) w(\boldsymbol{x}) d\boldsymbol{x} \approx \sum_{i=1}^{m} W_i f(\boldsymbol{x}_i) \tag{22}$$

where W_i and $\boldsymbol{x}_i$ are the multidimensional GGC weights and nodes, respectively, f is an integrand defined over Ω, and w is the pdf weighting function. As in the one-dimensional GGQ the GGC is designed so that Eq. (22) is exact for all functions for a pre-selected set. Classical choices of the pre-selected set of functions include polynomials up to a certain degree, trigonometric functions, and basis functions of a particular function space defined on Ω. Then the GGQ equation is written as

$$\begin{pmatrix} \int_{\Omega} \varphi_1(\boldsymbol{x}) w(\boldsymbol{x}) d\boldsymbol{x} \\ \int_{\Omega} \varphi_2(\boldsymbol{x}) w(\boldsymbol{x}) d\boldsymbol{x} \\ \vdots \\ \int_{\Omega} \varphi_m(\boldsymbol{x}) w(\boldsymbol{x}) d\boldsymbol{x} \end{pmatrix} = \begin{pmatrix} \varphi_1(\boldsymbol{x}_1) & \varphi_1(\boldsymbol{x}_2) & \dots & \varphi_1(\boldsymbol{x}_n) \\ \varphi_2(\boldsymbol{x}_1) & \varphi_2(\boldsymbol{x}_2) & \dots & \varphi_2(\boldsymbol{x}_n) \\ \vdots & & & \\ \varphi_N(\boldsymbol{x}_1) & \varphi_N(\boldsymbol{x}_2) & \dots & \varphi_N(\boldsymbol{x}_n) \end{pmatrix} \begin{pmatrix} W_1 \\ W_2 \\ \vdots \\ W_m \end{pmatrix} \tag{23}$$

where $\varphi_i(\boldsymbol{x})$ are the pre-selected functions and N is the function of functions used. Not that if the functions are selected to ℓ order polynomials in d variable then $N = N_\ell$. The equation above can be casted into a minimization problem to determine the weights and nodes as follows

$$J = \sum_{j}^{N} \left(\sum_{i}^{m} W_i \varphi_j(\boldsymbol{x}_i) - \int_{\Omega} \varphi_j(\boldsymbol{x}) w(\boldsymbol{x}) d\boldsymbol{x} \right)^2 \tag{24}$$

This work focuses on polynomials as the pre-selected set of functions so that $\varphi_1(\boldsymbol{x})$ are polynomial functions. From Eq. (23) it is seen that this system of equations has m multidimensional integrals that must be solved. Solving these integrals for high-dimensional space can be arduous. To simplify the process this work will focus on multi-dimensional Gaussian distributions. The benefit of Gaussian distributions is that the m multidimensional integrals in Eq. (23) have closed-form solutions.

The GGC for a Gaussian distribution can take a simple form if the $\varphi_j(\boldsymbol{x})$ functions are chosen to be Hermite polynomials for a multidimensional Gaussian distribution. The orthogonal polynomial for a Gaussian distribution can be written as a product of one-dimensional Hermite polynomials. To determine these polynomials, a coordinate transformation is used to covert the Gaussian distribution into a product of unit variate normal distributions. Specifically, given the following Gaussian distribution

$$p(\boldsymbol{x}) = \mathcal{N}(\mathbf{x};\, \boldsymbol{\mu},\, P) \tag{25}$$

the square-root factor S is computed from P (using a Cholesky factorization), and the following transformation is defined:

$$\mathbf{u} = S^{-1}(\boldsymbol{\mu} - \mathbf{x}) \tag{26}$$

Then the pdf of $\mathbf{u}$ can be written as

$$p(\mathbf{u}) = \mathcal{N}(u(1);\, 0,\, 1) \cdots \mathcal{N}(u(n);\, 0,\, 1) \tag{27}$$

where $u(j)$ denotes the j^{th} component of $\mathbf{u}$. Then the multidimensional orthogonal polynomials for variable $\mathbf{u}$ is written as the following product of polynomials:

$$P_{\boldsymbol{\alpha}}(\mathbf{u}) = \prod_{j=1}^{d} H_{\alpha_j}(u(j)) \tag{28}$$

where the subscript of the Hermite polynomial denotes the order of the one-dimensional polynomial. The multidimensional orthogonal polynomial is a product of univariate Hermite polynomials, given by

$$H_n(x) = (-1)^n \exp\left(x^2/2\right) \frac{d^n}{dx^n} \exp(-x^2/2) \tag{29}$$

where $H_n(x)$ is the orthogonal with respect to a one-dimensional unit-variate zero-mean Gaussian distribution. The mean of $H_n(x)$ is given by

$$E\{H_n(x)\} = \mu^n \tag{30}$$

where μ is the mean of the Gaussian weight function used to defined $H_n(x)$. From the definition in Eq. (26) it is noted that $\mathbf{u}$ has zero mean, and therefore $E\{H_n(u_j)\} = 0$ for $n = 0$ and $E\{H_n(u_0)\} = 1$ for $n \neq 0$. Define the following vector $\mathbf{L} \equiv [L_1 \ \ldots \ L_m]^T$, where

$$\begin{aligned} L_j &= \int_{\Omega} P_{\boldsymbol{\alpha}_j}(\mathbf{u}) p(\mathbf{u}) d\boldsymbol{u} \\ &= \int_{-\infty}^{\infty} \ldots \int_{-\infty}^{\infty} H_{\alpha_1}(u_1) \ldots H_{\alpha_n}(u_n) \mathcal{N}(u_1;\, 0,\, 1) \ldots \mathcal{N}(u_n;\, 0,\, 1) du_1 \ldots du_j \\ &= \int_{-\infty}^{\infty} H_{\alpha_1}(u_1) \mathcal{N}(u_1;\, 0,\, 1) du_1 \ldots \int_{-\infty}^{\infty} H_{\alpha_n}(u_n) \mathcal{N}(u_1;\, 0,\, 1) du_n \\ &= \prod_{i=1}^{d} E\{H_{\alpha_i}(u_i)\} \end{aligned} \tag{31}$$

Form Eq. (31) it can be seen that if $|\boldsymbol{\alpha}_j| \neq 0$ then $L_j = 0$. This simplifies the problem greatly since the N-dimensional integrals involved in Eq. (31) do not need to be calculated. Therefore, the following equation is true:

$$\mathbf{L} = \begin{pmatrix} \int_{\Omega} P_{\boldsymbol{\alpha}_1}(\mathbf{u}) p(\mathbf{u}) d\boldsymbol{u} \\ \int_{\Omega} P_{\boldsymbol{\alpha}_2}(\mathbf{u}) p(\mathbf{u}) d\boldsymbol{u} \\ \vdots \\ \int_{\Omega} P_{\boldsymbol{\alpha}_n}(\mathbf{u}) p(\mathbf{u}) d\boldsymbol{u} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{32}$$

Equation (23) can be written in matrix form by defining the following matrix:

$$M \equiv \begin{pmatrix} P_{\alpha_1}(\mathbf{u}_1) & P_{\alpha_1}(\mathbf{u}_2) & \ldots & P_{\alpha_1}(\mathbf{u}_n) \\ P_{\alpha_2}(\mathbf{u}_1) & P_{\alpha_2}(\mathbf{u}_2) & \ldots & P_{\alpha_2}(\mathbf{u}_n) \\ \vdots & & & \\ P_{\alpha_n}(\mathbf{u}_1) & P_{\alpha_n}(\mathbf{u}_2) & \ldots & P_{\alpha_n}(\mathbf{u}_n) \end{pmatrix} \tag{33}$$

Finding the GGC solution for the system of equations given in Eq. (23) involves finding both $\mathbf{w} \equiv [W_1, \ldots, W_m]^T$ and $\mathbf{z} \equiv [\mathbf{u}_1^T, \ldots, \mathbf{u}_m^T]^T$, which are combined into the vector defined by $\mathbf{\Theta} \equiv [\mathbf{w}^T \quad \mathbf{z}^T]^T$. The cubature rule is then given by the solution to the following minimization problem, $\mathbf{\Theta}_{\text{GGC}} = \min_{\mathbf{\Theta}} J(\mathbf{\Theta})$, where the objective function is given by

$$J(\mathbf{\Theta}) = \frac{1}{2} \left[\mathbf{L} - M(\mathbf{z})\, \mathbf{w}\right]^T \left[\mathbf{L} - M(\mathbf{z})\, \mathbf{w}\right] \tag{34}$$

Then the Jacobian matrix is calculated as $\frac{\delta J(\mathbf{\Theta})}{\delta \mathbf{\Theta}} = -\left[M(\mathbf{z}) \quad \Phi\right]^T \left[\mathbf{L} - M(\mathbf{z})\, \mathbf{w}\right]$, where Φ is given by

$$\Phi = \begin{pmatrix} w_1 \varphi_1^{'}(\mathbf{u}_1) & \ldots & w_n \varphi_1^{'}(\mathbf{u}_m) \\ \vdots & & \vdots \\ w_1 \varphi_m^{'}(\mathbf{u}_1) & \ldots & w_n \varphi_m^{'}(\mathbf{u}_m) \end{pmatrix} \tag{35}$$

Then $\varphi_j^{'}(\mathbf{u})$ terms are calculated by

$$\varphi_j^{'}(\boldsymbol{x}) = \begin{pmatrix} H_{\alpha_j}^{'}(\mathbf{u}_1(1)) \ldots H_{\alpha_j}(\mathbf{u}_1(d)) & \ldots & H_{\alpha_j}(\mathbf{u}_1(1)) \ldots H_{\alpha_j}^{'}(\mathbf{u}_1(d)) \end{pmatrix} \tag{36}$$

From the properties of the Hermite polynomials, $H_n^{'}(x)$ has a simple expression given by

$$H_n^{'}(x) = n H_{n-1}(x) \tag{37}$$

Then the minimization problem can be rewritten as follows

$$J(\mathbf{\Theta}) = \mathbf{w}^T K(\mathbf{z}, \mathbf{z}) \mathbf{w} - 2\mathbf{w}^T M^T L + 1 \tag{38}$$

where $K(\mathbf{z}, \mathbf{z}) \equiv M^T M$. Then the first-order optimality condition can be written as

$$\frac{\partial J(\mathbf{\Theta})}{\partial \mathbf{w}} = K(\mathbf{z}, \mathbf{z}) \mathbf{w} - M^T \mathbf{L} = \mathbf{0} \tag{39a}$$

$$\frac{\partial J(\mathbf{\Theta})}{\partial \mathbf{z}} = \Phi^T M \mathbf{w} = \mathbf{0} \tag{39b}$$

where $M^T \mathbf{L}$ is a vector of ones for Hermite polynomials. From Eq. (39) it can be noted that if $\mathbf{L} = M(\mathbf{z})\mathbf{w}$ the first-order optimality conditions are meet. Using the first-order optimality condition given in Eq. (39) the weight vector can be solved by

$$\mathbf{w} = K(\mathbf{z}, \mathbf{z})^{-1} M^T \mathbf{L} \tag{40}$$

Then substituting Eq. (40) into Eq. (38), the objective function can be written as

$$J(\mathbf{\Theta}) = 1 - \mathbf{L}^T M K(\mathbf{z}, \mathbf{z})^{-1} M^T \mathbf{L} \tag{41}$$

Equation (41) reduces the dimensionality of the minimization problem by n parameters. Then using Eqs (32) and (33) the objective function in Eq. (41) can be minimized, and a solution for the GGC can be found. This minimization does not require the evaluation of the N-dimensional integrals in $\mathbf{L}$, and can be vectorized for rapid evaluation of the objective function and the Jacobian. With Jacobian information efficient Newton-based methods can be used. These methods can be used to find Gaussian distribution cubature rules for different dimensions and order. More importantly, the definition of $\boldsymbol{\alpha}$ can used to find adaptive sparse rules to limit the growth of the number of points, and temper the curse of dimensionality.

COMPARISON OF CUBATURE METHODS

The GGC method is compared to both TPC and SGC method by integrating polynomial functions of two variables. The approaches discussed earlier for finding TPC, SGC, and GGC rules are used to find rules for distribution $\mathcal{N}\left(\mathbf{p} \equiv [x \;\; y]^T;\, \mathbf{0}_{2\times 1},\, I_{2\times 2}\right)$, and those points are shown in Figure 2. Then these points are used to approximate the following integrals:

$$I_{\alpha_x,\alpha_y} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^{\alpha_x} y^{\alpha_y} \mathcal{N}\left(\mathbf{p};\, \mathbf{0}_{2\times 1},\, I_{2\times 2}\right) dx\, dy \tag{42}$$

where α_x and α_y are the orders in the x and y dimensional, respectively. Then the integral in Eq. (42) is computed symbolically in closed-form for polynomial functions up to order 10. The closed-form solutions is then compared to the approximations given by TPC, SGC, and GGC rules for orders 3, 5, and 7. Figure 5 shows the resulting errors for the TPC, SGC, and GGC rules. From Figures 3(a), 3(b) and, 3(c) it is seen that the TPC rule captures the high-order polynomial integral for Eq. (42). This is expected since this rule is considered to use more points than is required for the designed order. Therefore TPC will also capture orders higher than its design order with the cost of requiring more points. For a two-dimensional example this might be a small drawback, but for higher-dimensional cases TPC rules are problematic since the number of points will increase drastically with dimensionality. The SGC approach overcomes this limitation by using a sparse tensor rule, which reduces the over-use of points. From Figures 3(d), 3(e), and 3(f) it can be seen that SGC does capture its designed order. Although the additional number of extra high-order polynomials required for SGC is less than TPC, it still captures the higher orders terms. Finally, from Figures 3(g), 3(h), and 3(i) it can be seen that the GGC approach is very precise in the orders that it captures, and it does a good job at integrating high orders. Also, its accuracy level has a sharper cutoff at its design order. This translates into an efficient number of points used for integration.

NUMERICAL RESULTS

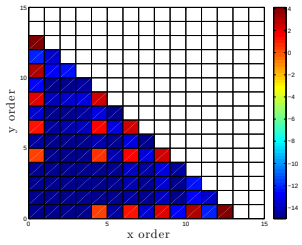
Fixed Transformation

An important test case for sigma point methods is the transformation related to conversion from polar to Cartesian coordinates. This transformation is used to highlight the utility of the Unscented Kalman Filter (UKF). The importance of this transformation is that it is used for observation models in many sensor scenarios, such as radar, line-of-sight, and laser range finding. The sensor will return bearing, θ, and range information, r, and this is converted to the target's position, $(x,\, y)$, is some global Cartesian coordinate frame. The transformation is given by

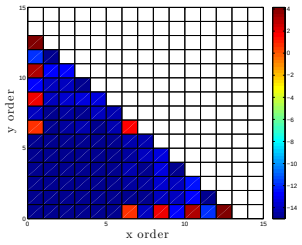
$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} r\cos(\theta) \\ r\sin(\theta) \end{pmatrix} \tag{43}$$

This transformation becomes especially difficult for large errors in bearing or large errors in θ. For this test case the mean of the polar coordinate position is set to be 0 degrees and 1 m, and the standard deviations are set to be $\sigma_\theta = 15$ degrees and $\sigma_r = 0.2$ m, respectively.

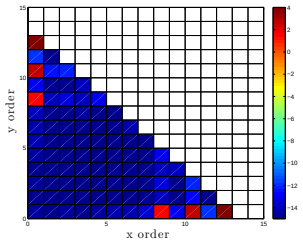
The results for the GGC are shown in Figure 3. Since the initial distribution in polar coordinates is Gaussian the Gaussian distribution GGC approach is used. Orders 3, 4, 5, and 7 are compared for accuracy. From Figure 3 it can be seen that as the order increases the mean and covariance estimates approach the Monte Carlo estimates. From this figure it can be seen that as the order increases more



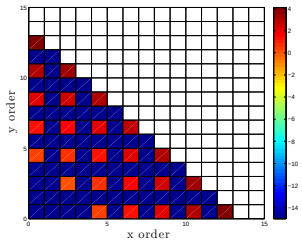
(a) Tensor GC Order 3



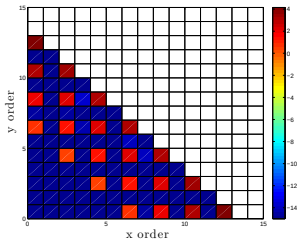
(b) Tensor GC Order 5



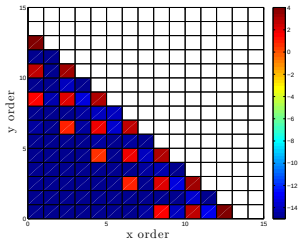
(c) Tensor GC Order 7



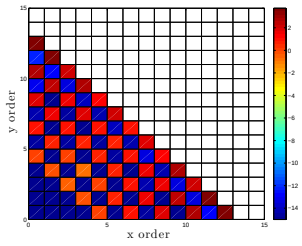
(d) Sparse GC Order 3



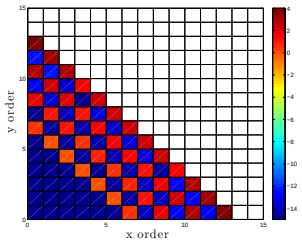
(e) Sparse GC Order 5



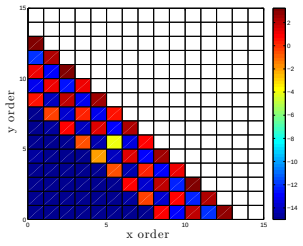
(f) Sparse GC Order 7



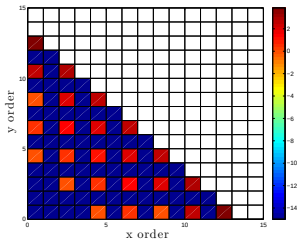
(g) GGC Order 3



(h) GGC Order 5



(i) GGC Order 7



(j) UKF

Figure 3. Polynomial Error vs Order in 2 Dimensions

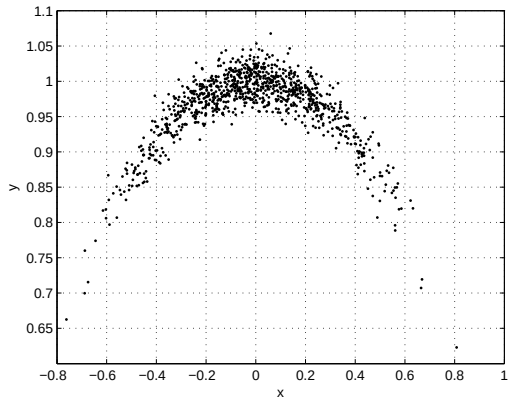


Figure 4. Monte Carlo Points for Polar to Cartesian Coordinates Transformation Example

nodes are used in the GGC approach, which captures the "banana" shape of the Monte Carlo points more closely.

Uncertainty Propagation: Nonlinear Spring Mass System

To test the performance of the new GGC rules for propagation of uncertainty through a dynamic system, a simple nonlinear two-dimensional example is used. This example models a nonlinear spring-mass system with a nonlinear friction term. The system dynamics for this simple example are written as

$$\dot{x}_1 = x_2 \tag{44a}$$

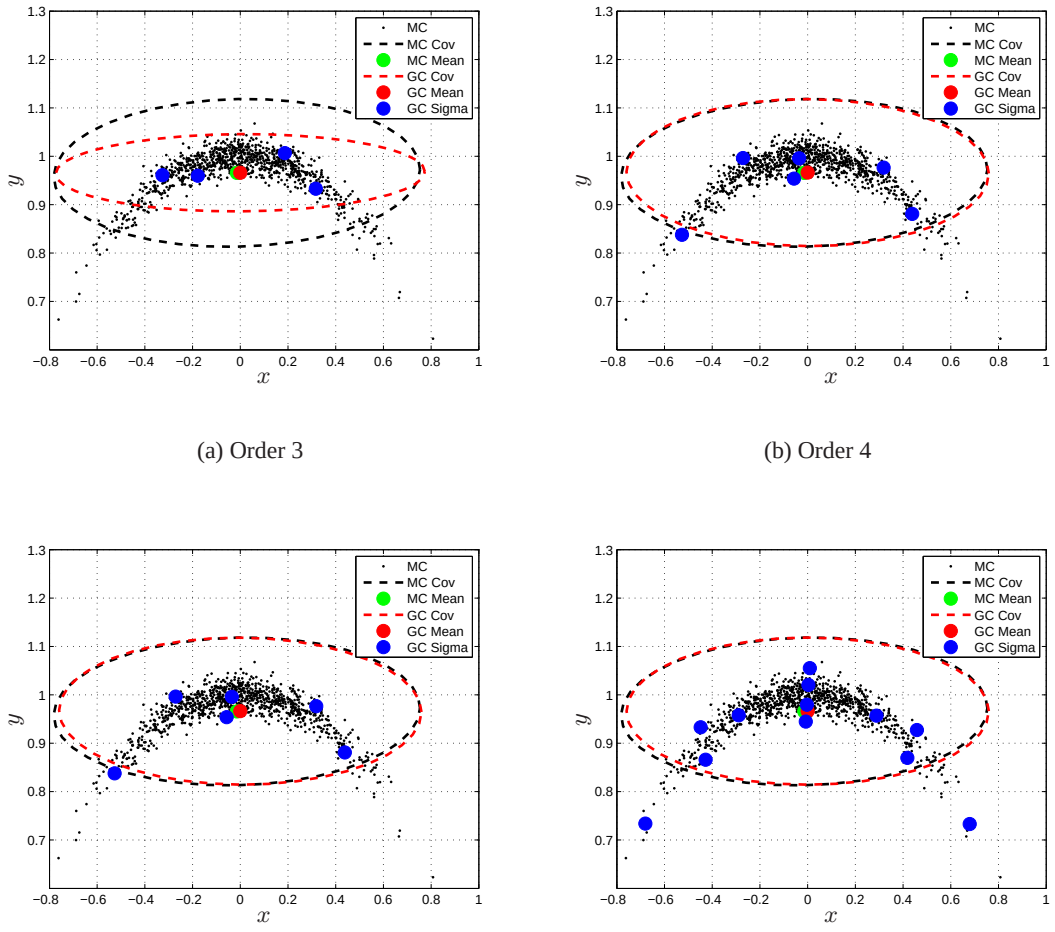
$$\dot{x}_2 = \kappa \dot{x}_1 + \varepsilon \dot{x}_1^3 + b\dot{x}_2 \tag{44b}$$

The parameters ε, κ, and b affect the system behavior to varying degrees. For this simulation these values are selected to be $\varepsilon = 0.1$, $\kappa = 1e^4/2\pi$, and $b = 0.05$. The simulation time considered is 200 seconds with a sampling interval 0.1 seconds. The initial state is given by $\mathbf{x}_0 = [0.2 \quad 0.3]^T$. The initial state covariance is given by $P_0 = \text{diag}([0.1^2 \quad 0.1^2])$. Monte Carlo simulations are conducted to test the GGC approaches. The Monte Carlo samples are taken from the initial distribution, and 1,000 samples are used in the comparisons.

The errors in the means x_1 and x_2 that are computed for the different order GGC are compared to the means calculated from the Monte Carlo samples at each time step. These errors are shown in Figures 7(a) and 7(b) for the x_1 and x_2 means, respectively. The norm of the state error is also calculated and is shown in Figure 7(c). From these figures it can seen that as the order increases the approximation of the mean is improved. During the initial portion of the simulation the errors are large for all the models since the nonlinear spring is fluxing greatly (as seen in Figure 6) but the higher-order GGC still has better performance. As the energy dissipates the system fluctuation is lower but the higher-order models still show better performance.

Example 1: Nonlinear Spring-Mass System

The first filtering example is a nonlinear spring-mass model with nonlinear measurements. The nonlinear spring-mass model is the same as the one used in Eq. (44) to test the uncertainty propa-



(a) Order 3

(b) Order 4

(c) Order 5

(d) Order 7

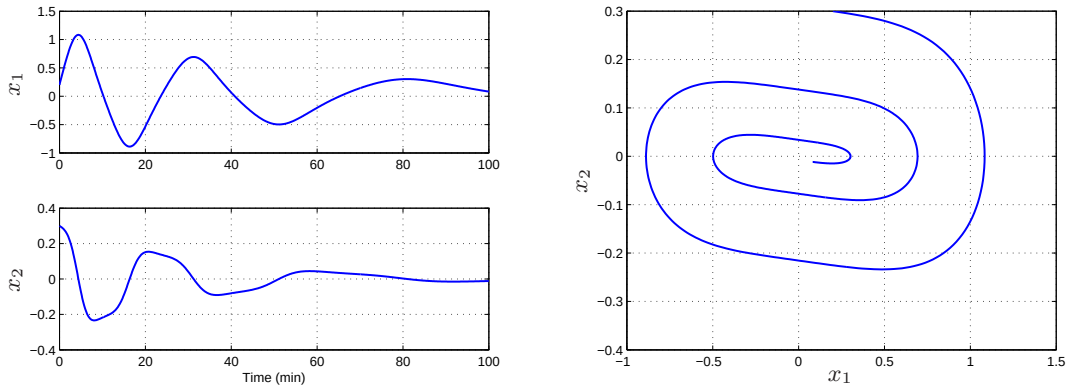
Figure 5. Polar to Cartesian Coordinates Transformation Example

gation performance of the GGC approach. The state variables are x_1 and x_2, which are the position and velocity of the spring-mass system. The following measurement function is used:

$$\tilde{y}_k = \exp\left(x_{1_k} - c\right) + v_k \tag{45}$$

where v_k is the measurement noise, which is defined to be $v_k \sim \mathcal{N}(0, \sigma^2)$, with $\sigma = 20$ and $c = 3$ for the simulation example. The initial true state used for this example is given by $\mathbf{x}_0 = [0.2 \quad 0.3]^T$. The initial state covariance is given by $P_0 = \text{diag}([0.1^2 \quad 0.1^2])$ and the initial state estimate is $\hat{\mathbf{x}}_0 = [0.3 \quad 0.4]^T$. Measurements are sampled at 0.1 second intervals for a total of 200 second simulation time. The measurement for this simulation scenario is given in Figure 9.

The state estimates for UKF, GGC order 3, and GGC 5 are shown in Figure 8. From Figure 8 it can be seen that the UKF has the worst performance while the GGC method does a better job at tracking the truth. All filters diverge initially, but the UKF has the largest divergence. Also the UKF takes the longest amount of time to converge to the true state.



(a) True States

(b) Phase Portrait

Figure 6. Numerical Integration Example Simulation Results

Example 2: Bearings Only Tracking

A classical filtering application in which a moving object is tracked by measuring only the bearings (angles) of the object with respect to the position of the sensors is shown here. There is one moving target in the scene and two angular sensors for tracking it. Solving this problem is important, because often more general multiple target tracking problems can be partitioned into sub-problems, in which single targets are tracked separately at a time. The state of the target at time step k consists of the position in two-dimensional Cartesian coordinates x_k and y_k and the respective velocities. The state vector can be expressed as

$$\mathbf{x}_k = \begin{pmatrix} x_k & y_k & \dot{x}_k & \dot{y}_k \end{pmatrix}^T \tag{46}$$

The dynamics of the target is modeled as a linear, discretized Wiener velocity model given by

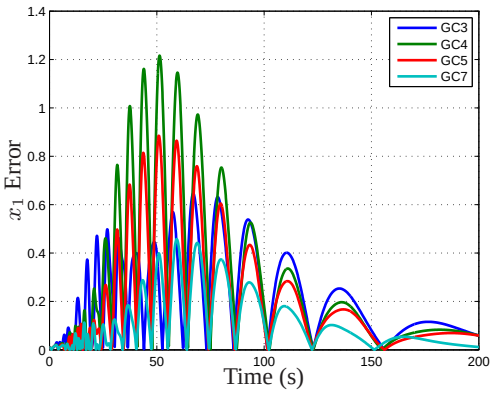
$$\mathbf{x}_{k+1} = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{k-1} \\ y_{k-1} \\ \dot{x}_{k-1} \\ \dot{y}_{k-1} \end{pmatrix} + \mathbf{q}_{k-1} \tag{47}$$

where $\mathbf{q}_{k-1}$ is Gaussian process noise with zero mean and covariance

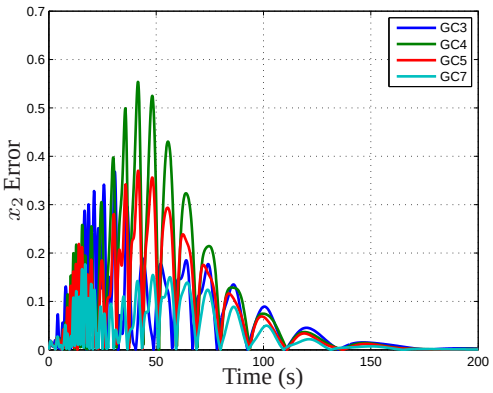
$$E\{\mathbf{q}_{k-1}\mathbf{q}_{k-1}^T\} = \begin{pmatrix} \frac{1}{3}\Delta t^3 & 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & \frac{1}{3}\Delta t^3 & 0 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 & 0 & \Delta t & 0 \\ 0 & \frac{1}{2}\Delta t^2 & 0 & \Delta t \end{pmatrix} q \tag{48}$$

where q is the spectral density of the noise, which is set to $q = 0.1$ in the simulations. The measurement model for sensor i is defined as

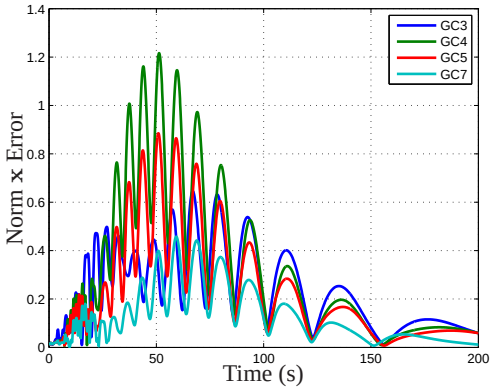
$$\theta_k^i = \arctan\left(\frac{y_k - s_y^i}{y_x - s_x^i}\right) + v_k^i \tag{49}$$



(a) x_1 Error



(b) x_2 Error



(c) Norm of x Error

Figure 7. Dynamic System Propagation Example

where $(s_x^i,\, s_y^i)$ is the position of sensor i and $v_k^i \sim \mathcal{N}(0,\, \sigma^2)$, with $\sigma = 0.05$ radians. Figure 10 shows a plot of a one realization of measurements in radians obtained from both sensors. The sensors are placed at locations $(s_x^1,\, s_y^1) = (-1,\, -2)$ and$(s_x^2,\, s_y^2) = (1,\, 1)$.

From the figure it can be seen that since the velocity errors are large all the models diverge initially. From Figure 8 it can be seen that the UKF has the largest divergence while the GGC approach has smaller divergence. Also, as the order increases in the GGC methods the divergence initially is less, and the filter does a better job tracking maneuvers.

Example 3: Attitude Estimation Example

This section discusses a simulation example that compares the performance of the GGC filter, UKF, and the EKF. The GGC uses the approach for representing the pdf as sigma points discussed earlier. Large initial errors are considered. It is important to note that with large initial errors the

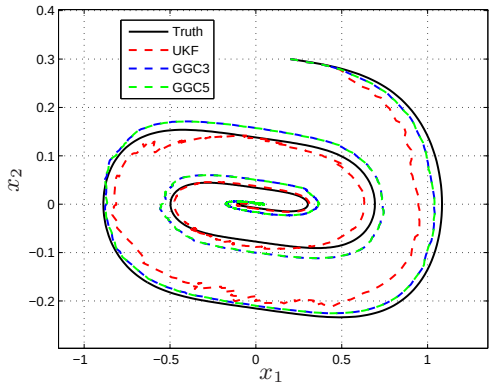


Figure 8. Nonlinear Spring State Estimation

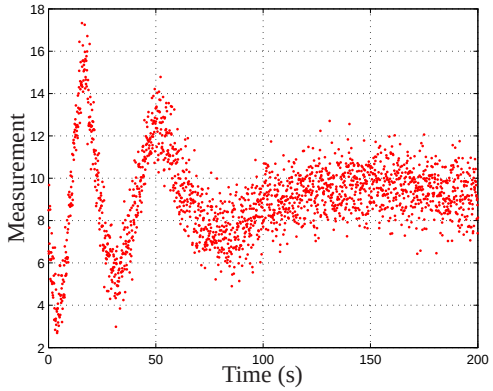


Figure 9. Nonlinear Spring State Measurements

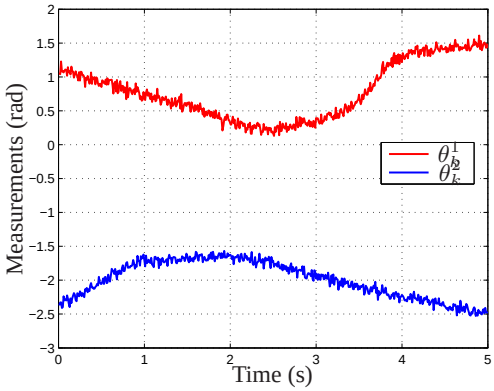


Figure 10. Bearings Only Tracking Measurements

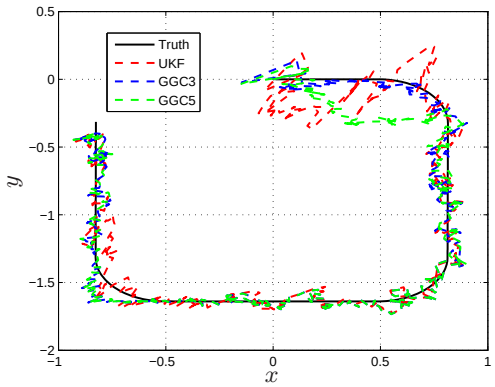


Figure 11. Bearings Only Tracking Filtering Example

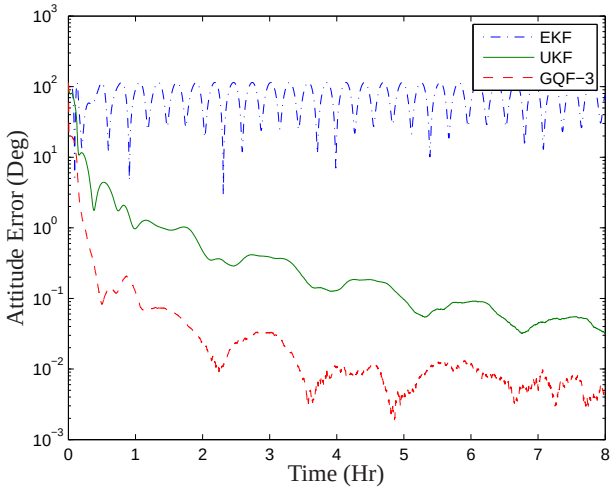


Figure 12. Attitude Error

attitude distribution becomes highly nonlinear.

The example shown here is taken directly from Ref. 2. Attitude errors of -50, 50 and 160 degrees for each axis, respectively, are added into the initial-condition attitude estimate. The initial attitude covariance is set to (50 degrees)^2 for each attitude component. The initial x- and z-axes estimates for the gyro biases are set to zero, however the initial y-axis bias estimate is 20 deg/hr. The initial bias covariance is set to (20 deg/hr)^2 for each axis. The initial particles are drawn using a Gaussian distribution with the aforementioned covariance matrices, which ensures that each filter is initialized in a consistent manner.

A plot of the norm of the attitude errors for this simulation case is shown in Figure 12. The EKF does not converge for this case since the first-order approximation cannot adequately capture the large initial condition errors. The UKF does have better convergence properties than the EKF for this case, however the GGC provides the best convergence performance. Both the EKF and the UKF

are more sensitive to the initial conditions than the GGC. The performance of the GGC is checked by running the filter 100 times. It always converges and the statistics of the estimation results in the 100 runs are almost identical.

CONCLUSIONS

This paper investigated a technique called generalized Gaussian cubature, where the rules are constructed to integrate a set of functions exactly. Hermite and Sparse Tensor products approaches for multi-dimensional integral were discussed. The generalized cubature approach was used to solve multi-dimensional integrals of Gaussian distributions. The solution was simplified greatly by using the generalized cubature technique along with properties of Gaussian distributions. Additionally, the solution of nodes and weights for the generalized cubature of a Gaussian distribution was also greatly simplified by using the multi-dimensional Hermite polynomials. Cubatures were derived for a number of orders and dimensions and used for both uncertainty propagation and sequential state estimation. Good results were shown with the new methods.

REFERENCES

[1] Kelecy, T. and Jah, M., "Analysis of High Area-to-Mass Ratio (HAMR) GEO Space Object Orbit Determination and Prediction Performance: Initial Strategies to Recover and Predict HAMR GEO Trajectories with No A Priori Information," *Acta Astronautica*, Vol. 69, 2011, pp. 551–558.

[2] Crassidis, J.L. and Markley, F.L., "Unscented Filtering for Spacecraft Attitude Estimation," *Journal of Guidance, Control and Dynamics*, Vol. 26, No. 4, July-Aug. 2003, pp. 536–542.

[3] Jia, B., Xin, M., and Cheng, Y., "Sparse-Grid Quadrature Nonlinear Filtering," *Automatica*, Vol. 48, No. 2, 2012, pp. 327–341.

[4] Arasaratnam, I. and Haykin, S., "Cubature Kalman Filters," *IEEE Transactions on Automatic Control*, Vol. 54, No. 6, 2009, pp. 1254–1269.

[5] Ito, K. and Xiong, K., "Gaussian Filters for Nonlinear Filtering Problems," *IEEE Transactions on Automatic Control*, Vol. 45, No. 5, 2000, pp. 910–927.

[6] Kalman, R.E. and Bucy, R.S., "New Results in Linear Filtering and Prediction Theory," *Journal of Basic Engineering*, March 1961, pp. 95–108.

[7] Julier, S.J., Uhlmann, J.K., and Durrant-Whyte, H.F., "A New Approach for Filtering Nonlinear Systems," *Proceedings of the American Control Conference*, Seattle, WA, June 1995, pp. 1628–1632.

[8] Yarvin, N. and Rokhlin, V., "Generalized Gaussian Quadratures and Singular Value Decompositions of Integral Operators," *SIAM Journal on Scientific Computing*, Vol. 20, No. 2, 1998, pp. 699–718.

[9] Stroud, A.H., *Approximate Calculation of Multiple Integrals*, Englewood Chiffs, New Jersey: Prentice-Hall, 1971.

[10] Smolyak, S.A., "Quadrature and Interpolation Formulas for Tensor Products of Certain Classes of Functions," *Doklady Akademii Nauk SSSR*, Vol. 4, No. 240-243, 1963, pp. 111.

[11] Ma, J., Rokhlin, V., and Wandzura, S., "Generalized Gaussian Quadrature Rules for Systems of Arbitrary Functions," *SIAM Journal on Numerical Analysis*, Vol. 33, No. 3, 1996, pp. 971–996.