We thank the reviewers for the detailed comments and valuable feedback. We have paraphrased questions from the reviewers and provided answers. We are particularly thankful to the reviewers for pointing out typos, and providing numerous suggestions for improvement. We will fix these in our revised version.

Reviewer # 1: How fast does the size of the polynomial form $x(t)$ grow? **Reviewer # 3:** Apropos theoretical upper bound on the number of terms.

As reviewer # 3 points out, if we restrict ourselves to a degree at most d polynomial over m variables, the number of terms grows as $O((n|W|)^d)$ where n is the number of time steps, $|W|$ is the number of disturbance variables in the system and d is the degree limit. In our implementation, we substantially reduce this number by truncating terms whose bounds are below a threshold: we add these terms to the residual interval. This makes our polynomials smaller. We could increase this threshold tolerance to adjust the polynomial size further: we hope to report on these effects as part of our extended version of our evaluation.

Reviewer # 1: Wouldn't it make sense to consider orthogonal descriptors, i.e., cumulants instead?

This is a very nice suggestion, thank you! Though we do not use cumulants, we use a trick described in lines 268-272 to reduce the overhead of computing the first four central moments of the sum of the individual components $p_1, \ldots, p_j$. The first three central moments are directly computed by adding the individual central moments (because they coincide with the cumulants!!) but the fourth central moment formula is slightly more involved since it involves the pairwise products of variances. However, with cumulants, we can simply add them. This would lead to very nice computational savings, much like the FFT polynomial multiplication algorithm. The key, however, is to compute the cumulants of each p_j rapidly.

Reviewer #1: one usually wants to reason about the uncertainties during run-time and not ex-post. This is the very reason why scalable algorithms that rely on message-passing (e.g., all variants of Kalman filters) are prevalent.

This is a good observation and we will clarify in our revision as follows: Polynomial forms accumulate the effect of disturbances from the very first step to the current step. (Extended) Kalman filters can be seen as using degree 1 terms (affine terms) and "compressing the distribution" from time 1 to n-1 into a single multivariate Gaussian before tackling time n. In some sense, our approach naturally lends itself to this kind of message passing if one is willing to approximate the polynomial form by a distribution at time $n-1$ before tackling the state update. Note that in filtering, there is the uncertainty propagation and the conditioning part. Conditioning on observations is not in scope for this paper but nevertheless an interesting problem for future work.

Reviewer # 2: Comparison with work of Scott Ferson et al. and the p-box method [12]

A p-box is a discretization of the joint distribution over the state space into cells with upper and lower bounds for the measure in each cell. Ferson's approach uses p-boxes as its representation of probability distribution. It allows us to answer queries about assertions and moments by summing up the contribution from each tile. As one may imagine, beyond 2 or 3 dimensions the number of tiles needed to maintain an accurate enough p-box representation to answer probability queries effectively can become prohibitive. Note that a discretization of the disturbances at each step is needed as well and the approach will effectively do an expensive convolution over all the tiles for the state variables and disturbances. Also note that five of the larger benchmarks in Table 1 have 5-10 state variables.

Reviewer # 2: Comparison with Bouissou et al. **Reviewer # 2**: Question about comparison with related work.

Please see Table 1 for the comparison with Bouissou et al. As you can see, polynomial forms provide much tighter bounds for moment and probability queries. Although one pays the price for using polynomials and higher order terms, polynomial forms are even faster in a few instances: this is because affine forms of Boussiou et al model nonlinear terms by fresh noise symbols, tracks correlation between the noise symbols and even maintains an ever growing covariance matrix involving the noise symbols. All of these are avoided in our polynomial forms approach.

Reviewer # 3 Are there not other techniques you could have compared with other than the affine method?

The only other method we could have compared apples to apples is perhaps the approach of Ferson et al with suitable alternations to fit our problem setup. However, we do not expect p-boxes to be scalable or precise (see response to Reviewer # 2 on comparison with Ferson et al. [12]). There are methods that are based on Monte-Carlo simulations, linearization of the dynamics, truncation and such: although we have cited some of these, we did not compare against them since those techniques have different formal guarantees. We have provided results from 10^5 simulations for reference.