Implementing the Matrix Exponential Function on Embedded Processors

James Lebak (presenter) and Andrea Wadell
MIT Lincoln Laboratory*
April 22, 2004

The solution to a differential equation of the form

\[ \dot{x}(t) = Ax(t), \quad x(0) = x_0 \]

is the function \( x(t) = e^{At}x_0 \) [5]. The expression \( e^{At} \) is the matrix exponential function. Examples of such equations arise in control theory and tracking applications.

A key application is the tracking of a ballistic target using noisy measurements. In this case, the matrix \( A \) above is actually a non-linear function of both \( x \) and \( t \). The extended Kalman filter (EKF) has been used in these tracking applications [1, 2]. The typical formulation of the EKF uses a first or second-order approximation to the solution of the differential equation to save operations [3]. While such implementation is efficient, it has been shown that in some conditions the EKF may show significant bias in altitude and ballistic coefficient [6]. Under such conditions it may be preferable to use the matrix exponential function directly.

In this paper we describe and benchmark an implementation of the matrix exponential function. The implementation is based on the standard technique of “scaling and squaring” from the literature [4, 5]. The major kernels in this technique are matrix multiplication and Gaussian elimination. In the matrix multiply kernel, the implementation makes use of SIMD vector extensions present on the PowerPC G4 (Altivec) and the Intel Xeon (SSE-2). Although the use of the matrix exponential expands the operation count of the extended Kalman filter substantially, benchmarks of the implementation show that the workload is well within the capabilities of modern processors.

References


*This work sponsored by the Department of the Navy under Air Force Contract F19628-00-C-0002. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Government.