grid are computed simultaneously. This drastic drop in complexity is somehow comparable to that achieved by the 'Fast Fourier Transform' in its domain.

Surprisingly, the key to this approach lies in the generalization to the non-linear case of a popular method which is widely used to compute numerically the exponential of a square matrix.

6.3.1 Matrix Exponential and the 'Scaling and Squaring' Method

The matrix exponential of a square matrix can be computed numerically in a large number of ways, with more or less efficiency [Moler 78]. One of the most popular of these numerical recipes is called the 'Scaling and Squaring' method, which is for example used by Matlab ™ to compute matrix exponentials [Higham 05]. Fundamentally, this method is very efficient because it takes advantage of the very specific algebraic properties of matrix exponential, which are in fact quite simple, as we shall see now. For any square matrix M, we have:

$$\exp(M) = \exp\left(\frac{M}{2}\right).\exp\left(\frac{M}{2}\right) = \exp\left(\frac{M}{2}\right)^2. \tag{6.11}$$

This comes from the fact that M commutes with itself in the sense of matrix multiplication. Iterating this equality, we get for any positive integer N:

$$\exp(M) = \exp\left(\frac{M}{2^N}\right)^{2^N}, \tag{6.12}$$

Then, the key idea is to realize that the matrix exponential is much simpler to compute for matrices *close to zero*. In this situation, one can for example use just a few terms of the infinite series of exponential, since high-order terms will be completely negligible. An even better idea is to use Padé approximants, which provide excellent approximations by rational fractions of the exponential around zero with very few terms. For more (and recent) details on this topic, see [Higham 05].

The 'Scaling and Squaring' Method for computing the matrix exponential of a square matrix M can be sketched as follows:

1. **Scaling step**: divide M by a factor 2^N, so that $\frac{M}{2^N}$ is close enough to zero (according to some criterion based on the level of accuracy desired: see [Higham 05] for more details).

2. **Exponentiation step**: $\exp\left(\frac{M}{2^N}\right)$ is computed with a high accuracy using for example a Padé approximant.

3. **Squaring step**: using Eq. (6.12), $\exp\left(\frac{M}{2^N}\right)$ is squared N times (only N matrix multiplications are required.) to obtain a very accurate estimation of $\exp(M)$.

In the rest of this Section, we will see how one can generalize this method to compute with an excellent accuracy polyaffine transformations based on autonomous ODEs.

6.3.2 A 'Scaling and Squaring' Method for LEPTs

Goal of the Method. We would like to compute efficiently and with a good accuracy the values of a Log-Euclidean polyaffine transformation at the vertices of a regular n-dimensional (well, 2D or 3D in practice) grid. The method described below will be referred to as the 'Fast Polyaffine Transform' (or FPT) in the rest of this Chapter.