

Pricing financial instruments more efficiently

13.02.2003

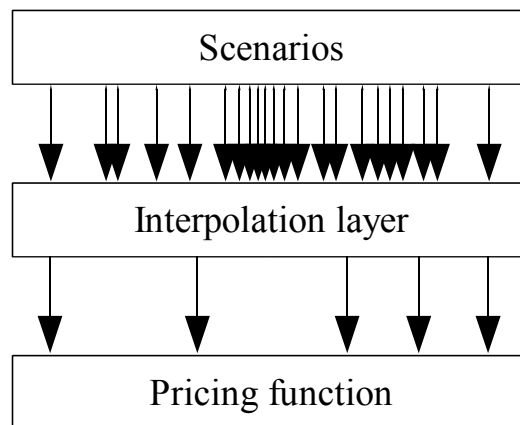
S.Dirnstorfer

Pricing financial instruments more efficiently

This document describes how the evaluation of a slow function, e.g. Monte Carlo simulation, can be considerably accelerated by using an interpolation scheme that minimizes the number of calls to the original function.

Vision

In every scenario a pricing function has to be evaluated. Since this function is usually extremely smooth it can be accurately approximated by interpolating only few evaluations.



Why Interpolation

A common method to approximate a pricing function facilitates the Greeks and is nothing else but a Taylor expansion. Although this approximation is optimal close to the spot position the error increases as risk factors become more extreme. However extreme risk factor scenarios are the crucial ones to determine value at risk computation. An interpolation of the pricing function can be accurate on the entire domain.

Taylor series, as they are computed by the Greeks, delta and gamma, approximate the function close to the current spot position.	By interpolating certain evaluated points, accuracy is far superior at the extreme points, as they are crucial in extreme scenarios.

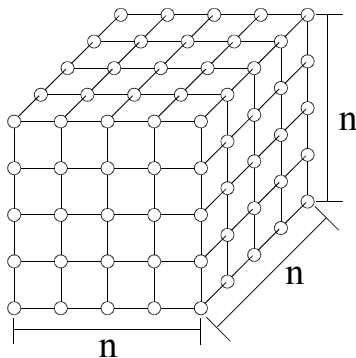
Curse of Dimension

A common problem of traditional interpolation scheme is the “curse of dimension”, which states, that the amount of evaluation points required to yield a certain accuracy increases exponentially with the

number of risk factors that determine the instrument's price. Consider for example a high dimensional point set which is equidistantly spaced in each direction. With D dimensions and n samples per risk factor one requires

$$n^D$$

function evaluations.



Example:

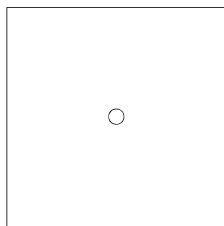
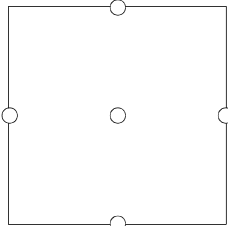
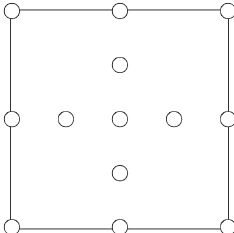
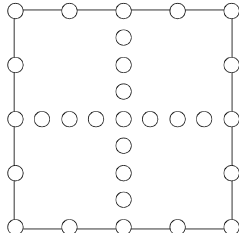
Assume a swap option depending on 10 risk factors, e.g. several interest rates tenors and an fx rate. Using only 2 points per dimension will require

$$2^{10} = 1024$$

evaluations, while only reproducing linear effects.

Sparse grids

Sparse grids have been proved to be the optimal point set for the interpolation of high dimensional smooth functions. The total number of function samples grows only slowly with the number of risk factors and the number of samples per risk factor.

1 st level		2 nd level	
# points = 1		# points = 1 + 2D	
3 rd level		4 th level	
# points = 1 + 3.5D + 0.5D ²		# points = 1 + 7.3D + 0.5D ² + 0.16D ³	

Example:

Assuming again 10 dimensions and a grid of level 3 will require

$$1 + 35 + 50 = 86$$

points and uses higher order polynomials to recover the function.

Combination method

The combination method is simple way of computing an interpolation on a sparse grid based on one dimensional interpolation schemes.

Let $Q^{(1)}$ be a unidimensional interpolation rule of level 1 using n_1 basis functions ϕ_k .

$$(Q^{(1)} f)(x) = \sum_{k=1}^{n_1} \phi_k(x) f(x_k)$$

The high dimensional sparse grid interpolation can easily be described by a tensor product of univariate interpolations.

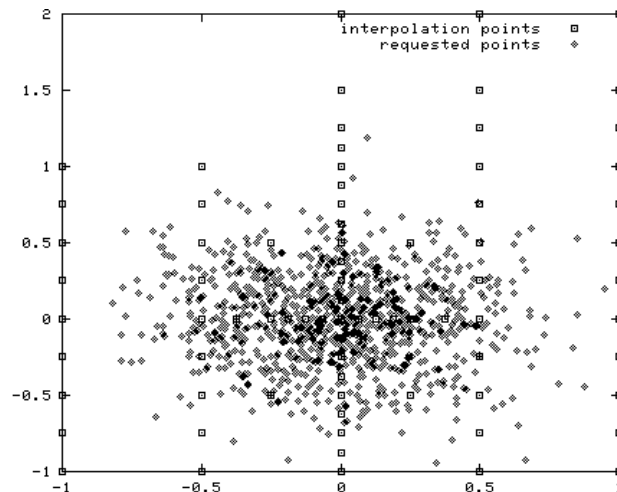
$$Q_L^{(d)} = \left(\sum_{k=1}^L (Q_k^{(1)} - Q_{k-1}^{(1)}) \otimes Q_{L-k}^{(d-1)} \right) f$$

The traditional full grid, which requires far more points, is generated as

$$Q_L^{(d)} = Q_L^{(1)} \otimes \dots \otimes Q_L^{(1)}$$

Adaptivity

If the function has different sensitivities to the risk factors, or behaves differently for certain combination of risk factors an adaptivity scheme can detect non smooth regions and uses more dense points where they are required.



Time dependency

Since pricing functions are not only smooth with respect to their risk factors, but also change slowly in time, the evaluated points can be reused to interpolate prices for consecutive days. This requires persistent data storage between two runs.

Conclusion

The presented technique minimizes the number of required function calls with minimal adjustments to the existing infrastructure. No pricing functions have to be touched and no interfaces have to be rewritten.

We expect speed ups in the order of factor 5 to 10 and implementation to be completed on very short time scale.

References

1. S. A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math. Dokl.*, 4:240-243, 1963. Russian original in Dokl. Akad. Nauk SSSR, 148 (1963), pp 1042-1045
2. M. Griebel, J.Garcke and M. Thess. Data mining with sparse grids, 2000.
3. S. Dirnstorfer. Numerical Quadrature on sparse grids, Technische Universität München, 2000