Universal Algorithmic Differentiation™

FINCAD

Unparalleled Risk Speed, Accuracy, and Coverage

Calculation speed, accuracy, and coverage are paramount concerns when calculating market risk for a portfolio or individual trade. FINCAD's F3 Platform provides an innovative approach to calculating market risk.

Our patented technique, as fully implemented in F3's Universal Algorithmic Differentiation™ (UAD™), provides unrivaled performance gains over traditional and slower methods still commonly deployed for risk calculations.

Overview

Calculation speed, accuracy and coverage are paramount concerns when calculating market risk for a portfolio or individual trade. FINCAD's F3 Enterprise Valuation and Risk Platform provides an innovative approach to calculating market risk. Our patented technique, as fully implemented in F3's Universal Algorithmic Differentiation™ (UAD™), provides unrivaled performance gains over more traditional and slower methods still commonly deployed for risk calculations. Furthermore, the technique does not require the relevant risk factors to be identified up-front using detailed model knowledge, but rather automatically calculates all appropriate sensitivities.

To date, many market participants continue to use methods such as bumping curves or market data points to assess the impact of market movements on their portfolios. For intra-day calculations or portfolios of increasing size and complexity, the performance drawbacks of such an approach becomes quickly apparent. These finite difference techniques require a revaluation of the portfolio for every risk factor, resulting in a linear scaling of calculation time with the number of risk factors. This situation often makes a fully comprehensive analysis of portfolio risk in real-time prohibitive. Rather, in the interest of time efficiency, intuition about the most likely relevant exposures is often relied upon.

Sophisticated sell-side institutions since the mid 90's have been using alternative techniques involving the calculation of first order sensitivities via partial derivatives. However, it is only relatively recently that this has been openly publicized. Under the broad banner of Automatic Differentiation (AD), these methods avoid the linear scaling of calculation time by using analytical differentiation and the Chain Rule of differential calculus. However, they present their own difficulties in terms of the complexity in retrofitting these methodologies to legacy codebases, as well as storage and other potential run-time inefficiencies. Furthermore the implementation of AD on a truly generic basis, for any conceivable trade, is a significant challenge.

Leveraging the same underlying mathematics as AD, FINCAD has developed UAD™ using advanced methods of software engineering. The architecture is fully described in an available technical paper. UAD™ provides significant gains in calculation efficiency as well as automatically ensuring that sensitivity calculations are fully comprehensive and truly generic.

Simply put, if you can value a trade or portfolio of trades in F3, then you are guaranteed to be able to calculate the first order sensitivity to all market data quotes which impact the valuation. There is no need to manually identify the relevant risk factors. This approach decreases calculation speeds by orders of magnitude over curve bumping methods. The guarantee applies for any mathematical model, for any trade (no matter how structured or complex), via any supported valuation approach (including Monte Carlo, closed form, or Backward Evolution).

This generic aspect sets F3 apart.

Analytical Sensitivities vs. Finite Difference

Below we briefly describe the difference between analytical sensitivities with Universal Algorithmic Differentiation™ and prevailing bumping methods.

Write $V(S_1, S_2, ..., S_N)$ as the value of a portfolio as a function of the N market quotes on which its value depends, S_i. Then $\Delta_i = \frac{\partial V}{\partial S_i}$ is the partial derivative, or "exposure", of V with respect to S_i. Computing Δ_i is traditionally achieved by the method of finite differences, or curve bumping. For some small (often one basis point) bump size, δS_i, Δ_i is commonly approximated based on the forward difference between the portfolio value at each point, as

$$\Delta_i = \frac{V(s_1, s_2, ..., s_i + \delta s_i, ..., s_n) - V(s_1, s_2, ..., s_i, ..., s_n)}{\delta s_i} + O(\delta s_i^2).$$

The crucial point is that the computational cost and calculation time of this approach scales linearly with the number of risk factors of interest. For example, if $N = 100$, then 101 valuations are required to obtain the exposure to every market quote, and if $N = 200$ then 201 valuations are required. For all but the smallest vanilla portfolios, this means that exposure to every relevant quote is seldom calculated in practice. Rather, approaches such as bumping an entire subset of market quotes ("bumping the yield curve") are followed. Another consequence is that some quotes can be ignored and relevant exposure missed because intuition, not computation, is used when exploring exposure. Alternatively this inefficiency in traditional curve bumping techniques is often remedied through the deployment of large and expensive dedicated hardware arrays.

In contrast to this brute-force approach to exposure calculation, it is possible to compute Δ_i exactly, at a computational cost that is essentially constant with respect to the number of risk factors, by applying the Chain Rule. An important point here is that a closed form solution for the pricing of trades in the portfolio under consideration is not required; this exact approach can also b e used in numerical computation such as Monte Carlo. This represents a significant advance over the curve bumping status quo, resulting in potentially several orders of magnitude of computational speedup. However, the approach is often not implemented in practice, or at best implemented in an ad-hoc basis for specific trades, since it often requires a significant reengineering of a legacy codebase.

The foundation of analytical exposure is differential calculus. For the portfolio V, we can write

$$dV(s_1, s_2, ..., s_N) = \sum_{i=1}^{N} \frac{\partial V}{\partial s_i} ds_i \equiv \sum_{i=1}^{N} \Delta_i ds_i.$$

As a conceptual aid, consider an algorithmic approach to calculating the Δ_i by allowing a recursive differentiation via the Chain Rule to propagate through the entire calculation. The calculation is made up of a number of operations, and we will look at one link in that chain. Consider the vector function $\boldsymbol{h}$ formed by composing the functions $\boldsymbol{f}$ and $\boldsymbol{g}$, such that

$$\vec{y} = \boldsymbol{h}(\vec{x}) = \boldsymbol{f}(\vec{u}),$$

$$\vec{u} = \boldsymbol{g}(\vec{x}).$$

In other words, the outputs from $\boldsymbol{g}$ are inputs to $\boldsymbol{f}$. Then the Jacobian J of $\boldsymbol{h}$, which represents the matrix of partial derivatives of all outputs with respect to all inputs, is

$$J_{ij} = \left[\frac{\partial \boldsymbol{h}(\vec{x})}{\partial \vec{x}}\right]_{ij} = \frac{\partial y_i}{\partial x_j} = \sum_k \frac{\partial y_i}{\partial u_k} \frac{\partial u_k}{\partial x_j} = \sum_k \left[\frac{\partial \boldsymbol{f}}{\partial u}\right]_{ik} \left[\frac{\partial \boldsymbol{g}}{\partial x}\right]_{kj} = \sum_k A_{ik} B_{kj}.$$

A and B are the Jacobians of $\boldsymbol{f}$ and $\boldsymbol{g}$ respectively. The computation of analytical sensitivities is therefore possible by not only implementing the functions $\boldsymbol{f}$ and $\boldsymbol{g}$ (and so on), but also implementing the Jacobian of each step, and propagating the results through the entire calculation chain.

There are, however, different approaches to how this propagation is implemented in software. The exact implementation in F3 avoids operator overloading and circumvents the expensive requirement of storing the intermediate Jacobians. Rather, the exposures are projected onto a spanning basis corresponding to the underlying variables. See the technical paper, Universal Algorithmic Differentiation™ in the F3 Platform, for a full description. The final result of any exposure calculation is embodied by the pairing of Δ_i with S_i, which represent the market quotes that were used to build curves, surfaces or otherwise impact the valuation.

UAD™ vs. Automatic Differentiation

Both UAD™ and AD share a degree of commonality insofar as both are methods of implementing the Chain Rule. However, UAD™ leads to a number of computational efficiencies which provide significant advantages over other implementations of analytical risk. Here we highlight a few key differences.

In a modern programming language, once a numerical algorithm's data types are substituted with custom types supporting AD, and the core operators are overloaded for those types, differentiation is truly automatic; the numerical algorithm's code remains unchanged and no further work is needed to obtain its derivatives. The penalty for this automation, however, is performance, with challenges in both storage and computational time. In contrast, UAD™ explicitly requires derivatives to be implemented for each step in the calculation chain. The simple requirement that the interface for the exposure calculation is implemented for each step allows F3 to use the optimal granularity for the problem at hand, yielding implementations that are efficient in both memory and time.

Many AD tools are available which retrofit a differentiation capability to existing code. Whether based on operator overloading or source-code transformation, these tools are forced to work at the granularity of the expressions in the existing code. The choice of granularity available in UAD™ arises because it was conceived before development started on F3 - it was built in from the start. Both object-oriented programming techniques and a careful development process ensure that all future valuation functionality will also support UAD™.

Much of the AD literature describes techniques for computing partial derivatives (exposures) to a known number of independent variables (risk factors). This leads to exposure calculations whose structure is constrained by the set of quotes for a given valuation and results ultimately in systems that calculate exposure to a fixed number of quotes, or specific types of quotes only. In contrast, UAD™ not only computes exposure to the relevant set of risk factors, but it also identifies that subset of relevant risk factors from the entire set of market data provided to the calculation.

Papers in the AD literature almost exclusively consist of applications to specific problems, of calculating exposure to known risk factors in the context of a specific model and valuation methodology. A handful of papers conduct a more general discussion and, at the time of writing, a growing number of banks have publicly indicated that they are currently applying AD techniques in some systematic manner in their next-generation library development. However, the FINCAD technical paper is the first to introduce ideas that facilitate a truly generic implementation of analytical exposure calculation - a guarantee that analytical exposure is available for every valuation.

Universal Algorithmic Differentiation™ (UAD™) in FINCAD's F3 presents a method of calculating analytical exposure, the result of which is truly universal. Indeed, one of the unique differentiators of F3's UAD™ is the completeness of its implementation of analytic exposure. UAD™ was designed into F3 from the start, resulting in a mature, stable, comprehensive and efficient platform for analytic risk computations that is unique amongst analytics vendors and, to the best of our knowledge, unparalleled by any other commercially available analytics provider. UAD™ is available for all types of risk factors and in every valuation, whether individual trades, portfolios, or CVA on any type of trade. The technology supports a rich set of applications that leverage analytic exposure computation, such as calculating hedging notionals, hedging costs, generic gamma/convexity, and profit-and-loss attribution.