Risk Optimisation: Finding the Signal in the Noise*

Benedict Burnett† Simon O'Callaghan and Tom Hulme

May 26, 2016

Abstract

We introduce a new method of optimising the accuracy and time taken to calculate risk for a complex trading book, focusing on the use case of XVA. We dynamically choose the number of paths and time discretisation to target computational effort on calculations that give the most information in explaining the PnL of the book. The approach is applicable to both fast, accurate intraday pricing calculations as well as large batch runs. The results are demonstrated by application to a large XVA book, which demonstrates speed-ups comparable to those available via adjoint algorithmic differentiation, for a fraction of the implementation cost.

1 Introduction

The make-up of bank trading books can be highly variable. At one end of the spectrum are flow books, consisting of a fairly homogeneous set of quick-to-value trades. At the other end, we have XVA books, referencing a heterogeneous population of counterparties, each with different underlying trade populations. These trades, and hence the XVAs for different counterparties, will vary enormously both in the time needed for calculation as well as the magnitude of the risk they represent. Naïve approaches that run each counterparty and each risk in the same way fall foul of this heterogeneity, spending the bulk of the calculation time where it is not necessarily needed, wasting time and computational effort, and ultimately leading to increased internal and client costs.

In this article we focus on the computational challenge posed by these heterogeneous books, taking XVA as our archetype, although the ideas are equally applicable to books of exotic interest rate or equities trades, which show a similar variety of computational costs and risk magnitudes. The importance of XVA has been highlighted by the recent explosion in adjustment literature, ranging from CVA and FVA (e.g., Piterbarg, 2010; Burgard and Kjaer, 2013) to MVA and KVA (e.g., Green and Kenyon, 2015; Green et al., 2014). The high-dimensional nature of the problem means that computationally costly Monte Carlo techniques are the only practical approach (Green, 2015), and so there is a keen interest in approaches to optimise these calculations.

An increasingly popular way to accelerate the calculation of these books is via adjoint algorithmic differentiation (AAD) (Giles and Glasserman, 2006; Capriotti et al., 2011), which allows the full set of first-order risks to be calculated analytically in a time comparable to

*Disclaimer: This paper represents the views of the authors alone, and not the views of Barclays Bank Plc.

†ben.burnett@barclays.com

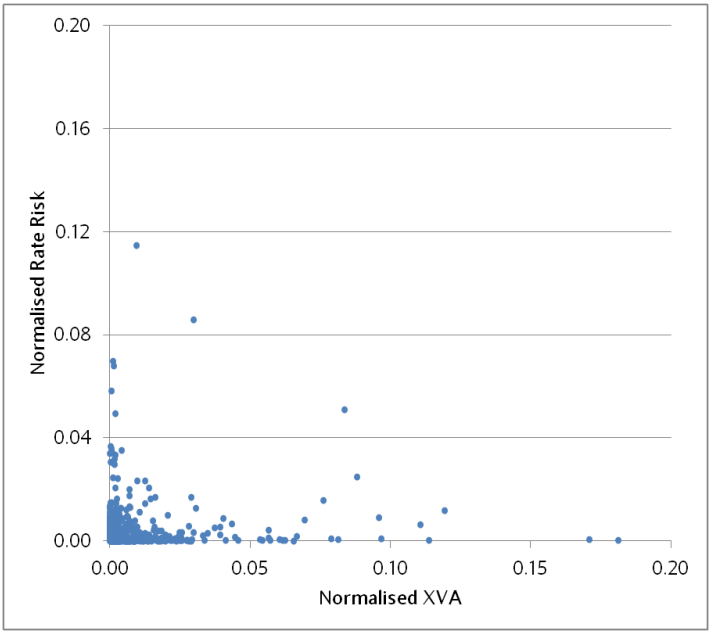

Figure 1: The normalised IR risk vs normalised PV of XVA for the book studied in section 3. We can see that for many counterparties, running the rate risk with the same simulation configuration as the baseline XVA will result in highly suboptimal calculations of the risk (either too precise, or too lax). A similar distribution is seen across other risk factors.

around five or six times that for calculation of just the 'present value' baseline simulation. AAD is rightly popular for its accuracy and speed, but has a few significant drawbacks – not least: (a) it requires significant re-engineering of the codebase, and (b) calculation of higher-order risks does not fit entirely naturally into its techniques (often requiring at least a partial fallback to finite-difference techniques).

AAD in itself does nothing to address the suboptimality mentioned above: each risk will be calculated on the same number of Monte Carlo paths as the baseline, frequently much more than a risk needs for reasonable accuracy. As an illustration, the distribution of risk magnitude vs present value magnitude is shown in fig. 1 for an example XVA book; we can see that there is extremely low correlation, suggesting that the risks requiring high path counts for decent accuracy are rarely tied to baselines with the same requirements.

In this article we suggest a technique which takes advantage of the very book heterogeneity cited above to massively speed up the calculation of the book by automatically focusing calculation time where it is most needed. We focus on minimising the unexplained PnL across a book: as well as the obvious applications to day-to-day trading activities, it is also a measure relevant to future FRTB regulations, whereby a desk is only eligible to use an internal model approach to calculate market risk capital if its PnL explain is sufficiently accurate (Basel Committee on Banking Supervision, 2013). The proposed technique can be applied with no re-engineering of the existing codebase (it can be implemented as a thin layer over the top), and allows a large book to be run with no manual configuration beyond a single input number that drives the batch's calculation time and overall accuracy. In our presentation, the technique provides gains comparable to those from AAD. For a codebase that already supports automatic differentiation however, our proposal can be used to improve

the baseline simulations, allowing AAD to provide the risks from these optimised calculations.

In this article, we introduce the theory of our approach by applying it to the total error produced from each batch calculation, and show some example results. We then show how the technique can also be used to accelerate intraday XVA pricing calculations, taking it beyond the batch-level applications described before.

2 Theory: Giving each risk the time it needs

The XVA desk in a bank standardly runs a nightly batch, calculating all the first- and second-order risks on the XVAs on each counterparty in its book. These risks are used to determine hedge amounts, to feed downstream risk management systems, and to perform the daily profit and loss (PnL) explain. This explain provides a good measure of the calculation accuracy across the book.

A trading desk is constrained in practice by the total compute capacity that is has available, and so it must make decisions on what risks to calculate and to what accuracy these should be computed. Third-order and higher risks are generally neglected, and all risks are necessarily calculated only to within a certain numerical error. These facts mean that every day there is unavoidably some amount of 'unexplain', measuring the extent to which the actual PnL differs from the expected amount, based on the calculated risks and market moves. In this paper we treat the overall PnL unexplain as the key quantity we want to minimise, as minimising this unexplain serves as a proxy for maximising the accuracy of the other associated calculations.

PnL explain is calculated via a Taylor expansion across the calculated risks and market moves. We assume each risk i has two main types of error: a standard error σ_i, measuring the finite precision due to Monte Carlo noise, and a bias β_i. For an XVA calculation, which is generally expressed as some sort of integral over a profile of simulated future exposures, the most obvious addressable source of bias comes from the granularity of simulation stopping dates: this results in missing convexity (due to approximating a curved profile with e.g. a piecewise linear sampling) and missed cash flow effects (i.e. smoothing over profile discontinuities). Standard implementations require sophisticated logic to incorporate major cash flow dates into the Monte Carlo; one of the benefits of the below logic is that significant flows automatically affect the stopping date density, so our algorithm can be applied without knowledge of these flow dates.

The two forms of error can be unified into a single global measure of the root mean square error (RMSE) η_i on each risk. For a risk x_i with 'true' value $\hat{x}_i$, we have that

$$\eta_i \equiv \sqrt{\mathbb{E}[(\hat{x}_i - x_i)^2]} \tag{1}$$

$$\equiv \sqrt{\sigma_i^2 + \beta_i^2}. \tag{2}$$

This will serve as our total error estimate on each result from a simulation. A risk's contribution to the day's unexplain will then be given by its RMSE η_i multiplied by the market move m_i in the corresponding risk factor or instrument.[1] If we crudely assume independence of all risks on all counterparties across the book, the errors on each risk and counterparty will add

[1]For a first-order risk, this market move is a simple concept. For a higher-order risk it is the product of the two corresponding first-order moves, scaled by a constant factor, following the terms in the Taylor series. For the baseline, we use $m_i = 1$ since its unscaled value directly affects the explain.

in quadrature to give the expected unexplain (the summation here and below runs over all risks on all counterparties):

$$\Psi^2 = \sum_i \eta_i^2 m_i^2. \tag{3}$$

Note though that when calculating the risks, we cannot know what the day's market moves m_i will actually be. What we can do is estimate the 'typical' daily move in each market factor – say 5 bp for an interest rate if we think rates standardly move by of this order over the course of a day. These typical move estimates need not be desperately precise. We let $\tilde{\eta}_i$ represent the product of η_i and its associated typical move, so that $\Psi^2 = \sum_i \tilde{\eta}_i^2$ (we use similar notation for other error types, $\tilde{\sigma}_i$ and $\tilde{\beta}_i$).

We now have our estimate of the day's unexplain from the batch. Our problem is to minimise this, with a constraint on the total calculation time. But this is just a classic optimisation problem, which can be addressed using a single Lagrange multiplier λ as follows (we constrain the total time to 1 without loss of generality). Letting π_i be the number of paths risk i is run on, ϕ_i be the frequency of stopping dates used for it, and τ_i be the CPU time it takes, we have two sets of equations:

$$\forall i: \quad 0 = \frac{\partial}{\partial \pi_i} \left\{ \sum_k \tilde{\eta}_k^2(\pi_k, \phi_k) + \lambda \left[\sum_k \tau_k(\pi_k, \phi_k) - 1 \right] \right\}, \tag{4}$$

$$\forall i: \quad 0 = \frac{\partial}{\partial \phi_i} \left\{ \sum_k \tilde{\eta}_k^2(\pi_k, \phi_k) + \lambda \left[\sum_k \tau_k(\pi_k, \phi_k) - 1 \right] \right\}. \tag{5}$$

To solve these we need to make some assumptions about the way $\tilde{\eta}_i$ and τ_i depend on π_i and ϕ_i:

1. For the standard error, it is simplest to assume standard Monte Carlo behaviour, such that this scales as $\tilde{\sigma}_i \propto 1/\sqrt{\pi_i}$. (This neglects deviations from orthodox standard error scaling if a quasi-random generator is used for the Monte Carlo.)

2. We assume the bias is a function only of the stopping date frequency ϕ_i. Experimentally a power law of the form $\beta_i \propto \phi_i^\alpha$ is a good fit to XVA problems, with α lying in the range [0.5, 1.5]. Thus we will take $\alpha = 1$ for the remainder of this paper to simplify the mathematics. The extension to general α is obvious.

3. We take the simulation time to be proportional to the product $\pi_i \phi_i$, since e.g. doubling the number of paths or dates will require twice as many trade valuations and lifecycle operations. (This neglects setup costs.)

Any preferred laws can be used in place of these simple assumptions, but the above suggestions provide a reasonable starting point.

Defining constants of proportionality such that $\tilde{\sigma}_i = \mu_i/\sqrt{\pi_i}$, $\tilde{\beta}_i = \nu_i/\phi_i$ and $\tau_i = \gamma_i \pi_i \phi_i$, eqs. (4) and (5) give us that

$$\forall i: \quad \lambda = \frac{\mu_i^2}{\gamma_i \pi_i^2 \phi_i} = \frac{2\nu_i^2}{\gamma_i \pi_i \phi_i^3}, \tag{6}$$

which simplifies to

$$\forall i: \quad \lambda = \frac{\tilde{\sigma}_i^2}{\tau_i} = \frac{2\tilde{\beta}_i^2}{\tau_i}. \tag{7}$$

Note that since λ is not a function of i, this equation enforces equality of the effective variance-to-time and squared-bias-to-time ratios across all risks, on all counterparties in the book.

Our remaining problem is that the constants of proportionality are unknown – and for a standard XVA batch (where trades come and go on a daily basis), they can be expected to vary significantly over time.

This difficulty can be addressed by a method of 'exploratory Monte Carlo', in which a lightweight initial simulation (with say 100 paths) is used as a means to calibrate the scaling relationships described earlier. Estimates of μ_i and γ_i follow fairly directly from measurements of this simulation's standard error and computation time, while ν_i follows from a calculation of the bias on the result. This bias can be estimated by comparing the XVA resulting from an exposure profile made up of all simulation dates (a 'high-frequency' XVA value) with the XVA resulting from using the same simulated paths but only a subset of simulation dates to define the exposure profile (a 'low-frequency' XVA).[2]

Specifically, taking the pre-simulation to be run on p paths and with stopping date frequency f, and letting the initial standard error on risk i be s_i, its bias be b_i and its calculation time be t_i, we can see that

$$\sigma_i = s_i \sqrt{\frac{p}{\pi_i}}, \tag{8}$$

$$\beta_i = b_i \frac{f}{\phi_i}, \tag{9}$$

$$\tau_i = t_i \frac{pf}{\pi_i \phi_i}. \tag{10}$$

Substituting these relations into eq. (7), we finally obtain:

$$\pi_i = p \left(\frac{\tilde{s}_i^6}{2\lambda^2 \tilde{b}_i^2 t_i^2} \right)^{1/5}, \tag{11}$$

$$\phi_i = f \left(\frac{4\tilde{b}_i^4}{\lambda \tilde{s}_i^2 t_i} \right)^{1/5}. \tag{12}$$

Note the significance of these formulae: without knowledge of any other risks or counterparties, we can select the optimal path count for counterparty and risk i based entirely on a quick initial simulation to find its (s_i, b_i, t_i) values each day.

The workflow we employ is therefore to take each counterparty in turn, and run it (baseline and risks) with settings (p, f); eqs. (11) and (12) then give the ideal numbers of paths/dates for each risk. The risks are rerun with these ideal settings to obtain the optimal usage of available computer power, before moving on to do a (p, f) run for the next counterparty in the batch.[3]

[2] This approach to estimating a realised bias on the result has much in common with the approach of 'multilevel Monte Carlo' techniques (see Giles, 2015, for a survey).

[3] The above formalism can be derived alternatively on information-theoretic grounds, by considering the mean squared error (MSE) on each risk in probabilistic terms across the batch – the fraction of MSE provided by each unit of time spent on each risk can be viewed as a probability. Maximum informational efficiency is achieved for maximum entropy, where all samples are providing information (MSE) at the same rate – as suggested by eq. (7). It can be seen that λ effectively represents a 'marginal information rate': it is proportional to the rate of reduction of MSE given by the last unit of computation time.

The only parameter that must be chosen by a user in this routine is λ. Any value of λ will provide *an* optimum; it becomes a question only of choosing *which* optimum is wanted. The different choices represent different trade-offs between global accuracy vs overall time: note that the total time constraint has dropped out of our final formulae, and is controlled entirely by the choice of λ. In our experience, a small number of trials running the batch with different values of λ are sufficient to home in on a suitable value, which can then be held constant.

The result is that we have now a 'global' optimiser, taking account of both standard error and bias simultaneously, controlled by a single human input λ that drives the whole batch.

2.1 Neglecting bias

The above derivation can also be followed neglecting the contribution of bias to the total error, thus focusing entirely on how the standard error and time scale with the path count, and seeking to minimise the total variance across the book with fixed stopping dates. This may, for example, be useful in those cases where we need to report all profiles on a common set of dates. In this case eq. (11) becomes

$$\pi_i = p\frac{\tilde{s}_i}{\sqrt{\lambda t_i}}. \tag{13}$$

3 Application

In this section we outline the effects of applying the above optimisation theory to an XVA book at Barclays.

3.1 Implementation

An important feature of the above theory is that it can be applied in a non-intrusive manner on top of any existing calculation of the XVA. If it is possible to provide the number of paths, stopping date schedule, and desired risks as part of the calculation configuration, then all that is needed for the optimisation is a separate lightweight component that can (a) understand the results of the calculation, (b) perform the simple mathematics of eqs. (11) and (12), and (c) trigger an updated simulation.

Given these calculations are far more lightweight than running a Monte Carlo, this 'optimiser' can be written as a standalone component in a high-level language, making it (a) very quick to write, (b) a very efficient means to perform further enhancements down the line at the same level, and (c) relatively easy to support.

There are several practical choices which can be incorporated in the concrete implementation: for example, the optimal number of paths and dates can be rounded to standard values opening up the optimisation possibilities that come from running multiple risks on the same settings (e.g. baseline reuse). Likewise, risks that depend only on the baseline simulated paths (e.g. CVA credit risks in a deterministic credit model) can be included as part of the simulation with the most stringent settings: this will often give these risks to a better-than-targeted final accuracy 'for free'. Multiple metrics can of course be run simultaneously; a scaling can be applied to render quantities such as RWAs comparable with XVAs in the optimisation logic.

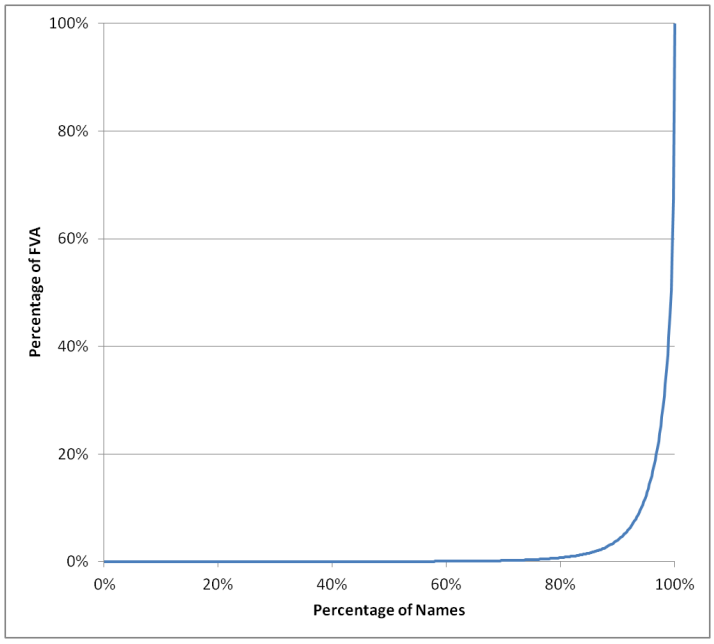

Figure 2: Lorenz graph of FVA contributions from counterparties across the batch. To illustrate the concentration, around 96% of the FVA comes from 10% of the counterparties.

3.2 Book statistics before optimisation

Figures 2 and 3 show the pre-optimisation statistics of a typical XVA book. We can see that the book is highly heterogeneous, with a large percentage of the XVA being calculated extremely quickly, followed by a long tail of calculation time expended on a very small residual. The calculation time per counterparty was almost entirely uncorrelated with the XVA (and error) size.

End-to-end wall time for calculation of the entire book took around 7 hours, despite being distributed to a large grid. It thus consumed most of each night's grid time.

3.3 Book statistics after optimisation

Application of the optimisation logic began with a small set of experiments on the batch to find a suitable value of λ to give acceptable accuracy in a reasonably short end-to-end time. Once this value had been found it was left static (though the value could be altered if the book statistics change significantly over time).

The effect on the batch was dramatic. The end-to-end time dropped to around one hour, with the total grid time being reduced by 94%. This speed-up was achieved whilst simultaneously improving the overall accuracy, such that the daily unexplain was reduced by a factor of around 3 by switching on the optimiser. A comparison against a high-path, high-density reference run showed the gross RMS 'true' error (i.e. deviation from the correct numbers) dropped by a factor greater than 10 when the optimiser was used.

Figure 3 shows the impact on the distribution of calculation times. We can see that the highly unequal distribution visible before the change is significantly softened by the application of the optimiser, resulting in a more well-behaved distribution closer to uniform time expenditure per unit of XVA. Note that this is a by-product of the fact that we target the RMSE, whose distribution is shown in fig. 4. These graphs focus on the baseline XVA; the statistics for risks are similar.

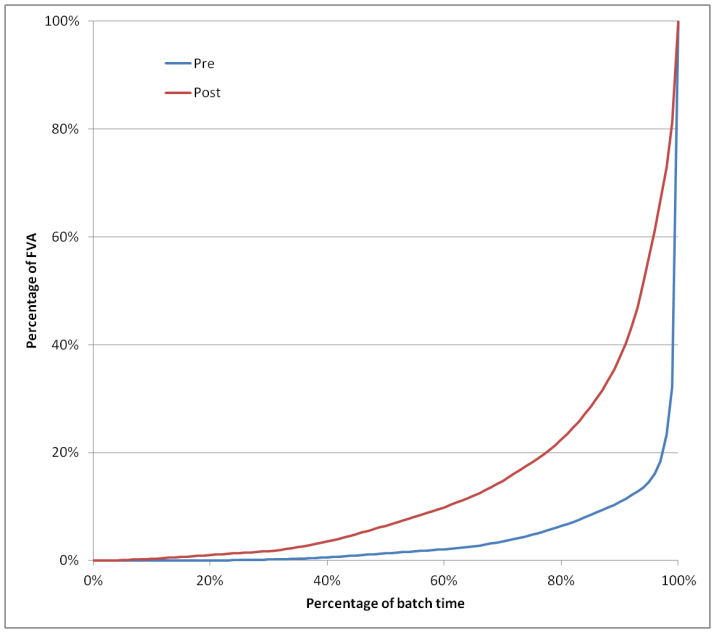

Figure 3: Lorenz graph of FVA contributions vs proportion of batch time (focusing on baseline calculation), before ('Pre') and after ('Post') the optimisation was applied. It can be seen that the racking-up of XVA contributions is a rather more steady function of grid time after optimisation than the tight dog-leg seen before (which implied that far too much time was being spent on small contributions).

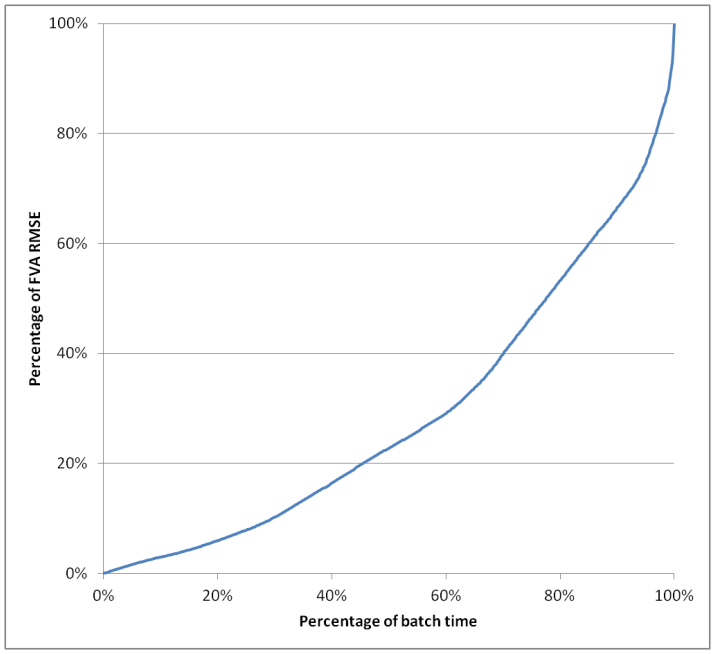

Figure 4: Cumulative FVA RMSE vs proportion of batch time after optimisation.

4 Extensions

4.1 Reducing risk counts

Various extensions are possible using the above philosophy of spending time where it is most needed. We have discussed this in the context of running only as many paths and dates as are needed; the same thinking can be applied to reduce the number of risks run in total. As an example, the PnL impact of a strip of tenor deltas for a curve is likely to be on the same order as that of the parallel shift delta, whose size can therefore be used as a trigger. Using the RMSE η_i as a gauge of the uncertainty on the parallel delta, a relatively conservative estimate of the PnL impact would therefore be $(|\Delta_i|+\eta_i)m_i$ for a parallel risk Δ_i. This quantity can be compared to a user-defined threshold to determine whether to run its full strip risk. Similarly, the PnL impact of a second-order risk can be heuristically estimated using a product of its underlying deltas, which can again be compared to a user-defined 'cross-gamma threshold'.

Such an approach is necessarily rather crude, but pays dividends: using it, we were able to remove around 50% of tenor and higher-order risk from the batch, with a sacrifice of 0.4% of gross daily unexplain. This has an obvious impact in terms of reducing the grid time needed for the batch run, but it also allows a significant reduction in IT time spent processing the huge number of results generated from each batch.

4.2 Intraday pricing

The same technique can be applied also to the use case of fast and accurate pricing of a new deal. Consider a trader needing to run the present value and significant risks on the XVA of a counterparty, and wanting a particular RMSE accuracy $\tilde{H}$ across all risks combined (the trader should not neglect either the standard error or the bias in this setup), but needing the calculation to be run as fast as possible.

This can be directly cast in the formalism from the preceding section. Using the fact that $\tilde{H}^2 \equiv \sum_i \tilde{\sigma}_i^2 + \tilde{\beta}_i^2 = \sum_i \frac{3}{2}\tilde{\sigma}_i^2$ from eq. (7), we obtain the required value of λ:

$$\lambda = \frac{4\tilde{H}^5}{\left[3\sum_i \left(\tilde{s}_i^2 \tilde{b}_i t_i\right)^{\frac{2}{5}}\right]^{\frac{5}{2}}}, \tag{14}$$

which can then be substituted into eqs. (11) and (12) to obtain the optimal path and date settings.

Similarly, if the trader instead has a requirement of 'the best possible accuracy achievable in a (grid) time of T', we obtain:

$$\lambda = \left[\frac{\sum_i \left(2\tilde{s}_i^4 \tilde{b}_i^2 t_i^2\right)^{1/5}}{T - \sum_i t_i}\right]^{5/3}. \tag{15}$$

5 Conclusions

We have presented the theory behind an effective method to improve calculation efficiency for heterogeneous trading books, along with a case study in its application. Our case study

focussed on XVA, but the technique should be effective for any exotic book with raw calculation statistics comparable to fig. 2. With low implementation cost, we can take advantage of inefficiencies in standard Monte Carlo setup for large books to achieve speed-ups comparable to those available by AAD. After optimisation, calculation of the risks on the XVA took only around 4–5 times longer than calculation of the optimised baseline values. AAD leaves the suboptimality visible in fig. 3 unaddressed, and would standardly involve running all the risks on the same path and date count as the baseline, neglecting the decorrelation visible in fig. 1.

The theory developed here can be extended in various ways. To take an example, although we have used a Monte Carlo setup as an illustration, it could be readily extended to PDE calculations given suitable assumptions about the scaling of errors with calculation time.

An obvious question about our implementation is whether the initial exploratory p-path simulation for each counterparty and risk need be performed every time the batch is run, or whether the statistics from a single initial run could not be used for numerous following batches to save time. In other applications that might be a reasonable approach, but the continual re-exploration we propose is well-suited to XVA books. It automatically copes with changing trade populations, moneyness, CSA, netting, etc, in a manner analogous to importance sampling: we sample more densely those calculations which contribute to batch-level unexplained PnL. The underlying assumptions are necessarily crude, but that very property renders the optimisation relatively robust to variations in the behaviour of the batch. It is also simultaneously applicable to a broad range of metrics: the full range of XVAs and RWAs provide an obvious starting point.

References

Basel Committee on Banking Supervision. Fundamental review of the trading book: A revised market risk framework. 2013. URL `http://www.bis.org/publ/bcbs265.pdf`.

C. Burgard and M. Kjaer. Funding Costs, Funding Strategies. *Risk*, pages 82–87, December 2013. Available at `http://papers.ssrn.com/abstract=2027195`.

L. Capriotti, J. Lee, and M. Peacock. Real Time Counterparty Credit Risk Management in Monte Carlo. *Risk*, pages 82–86, May 2011. Available at `http://papers.ssrn.com/abstract=1824864`.

M. Giles. Multilevel Monte Carlo methods. *Acta Numerica*, 24:259–328, 2015.

M. Giles and P. Glasserman. Smoking adjoints: fast Monte Carlo Greeks. *Risk*, pages 92–96, January 2006.

A. Green. *XVA: Credit, Funding and Capital Valuation Adjustments*. Wiley, 2015.

A. Green and C. Kenyon. MVA by Replication and Regression. *Risk*, April 2015. Available at `http://papers.ssrn.com/abstract=2432281`.

A. Green, C. Kenyon, and C. Dennis. KVA: capital valuation adjustment by replication. *Risk*, November 2014. Available at `http://papers.ssrn.com/abstract=2400324`.

V. Piterbarg. Funding beyond discounting: collateral agreements and derivatives pricing. *Risk*, pages 97–102, February 2010.