Efficient Monte Carlo simulation of portfolio value, value-at-risk and other portfolio metrics

Presentation to the Online Actuarial Colloquium 2022

By Malcolm Kemp
Managing Director, Nematrian

21 June 2022

Nematrian

Efficient Monte Carlo simulation

- Introduction
- Proposed simulation approach: tri-segmented Monte Carlo
- Illustrative exercise

Introduction (1)

- Financial firms (and their regulators) often need estimates of portfolio values or of risk measures such as Value-at-risk (VaR), expected shortfall, …
  - Sometimes these can be calculated analytically but more usually larger firms need to use simulation techniques. Similar picture in non-financial field.
- Traditional workhorse for this purpose is *Monte Carlo simulation*
  - In most basic form (equally probable) simulations are drawn randomly from relevant probability distribution characterising economic drivers impacting the (present) value of the (overall) portfolio payoff
  - Accuracy typically improves only in proportion to square root of number of simulations used, i.e. for accuracy ε requires $O(\varepsilon^{-2})$ sample draws
  - For large / complex books (especially with nested calculations), runtimes can be excessive to obtain an adequately low level of error

Introduction (2)

- Researchers have explored many ways of speeding up basic Monte Carlo including:
  - Antithetic variables (and other "moment fitting" techniques): e.g. if $S_+ = \mu + x$ included in simulation set then also include $S_- = \mu - x$
  - Control variate techniques: approximation $\tilde{P}$ to true payoff P is identified where $\tilde{P}$ is quick to calculate and we estimate e.g. $\mathbb{E}(P)$ as $\mathbb{E}(\tilde{P}) + \mathbb{E}(P - \tilde{P})$
  - Importance sampling (aka stratified sampling): preferentially draw samples from parts of underlying distribution expected to contribute most to error in the end answer, and adjust weights accordingly
  - Low discrepancy sequences: select points more uniformly across probability distribution space than would arise with pure random Monte Carlo draws
- Or throw (parallelised) computer resources (e.g. in the cloud), symbolic engines, quantum computers, … at the problem

Introduction (3)

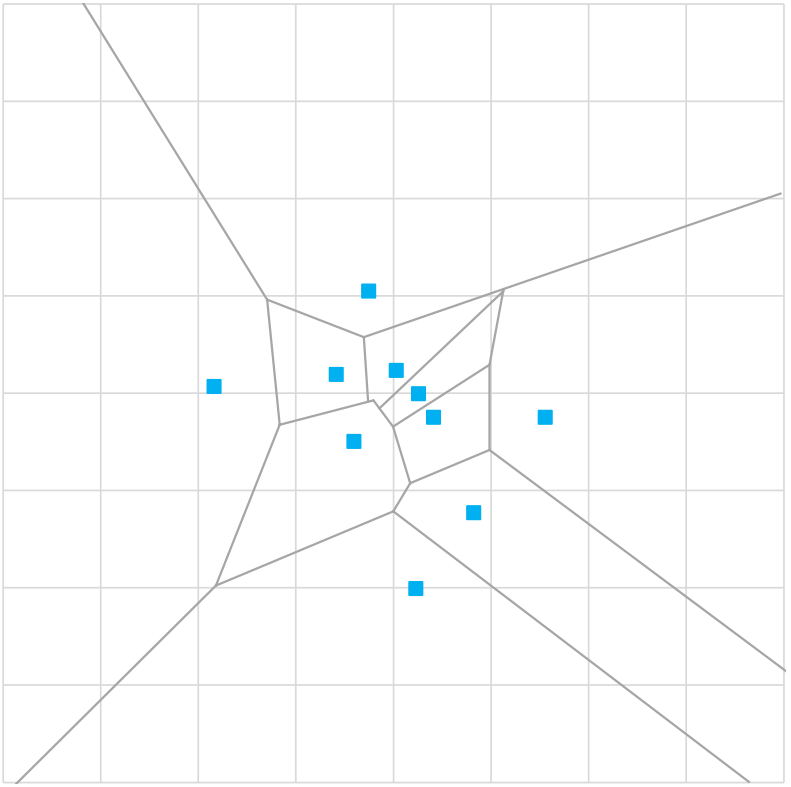
- Problem most acute when runtime cost of applying a given sample to overall portfolio is very large relative to runtime cost of drawing that sample from its (assumed given) underlying probability distribution
- Kemp (2019) “Improving valuation runtimes for derivative books” (Nematrian) proposed a “targeted quantile-spacing approach” which can address this:
  - Prepare a very large ‘extended’ simulation set, size N and use it to prepare a much smaller ‘collated’ simulation set, size n, with only the collated set actually applied to the portfolio
  - Simplest case, sort the N extended simulation set members and select for the collated set ones that are equally spaced in quantile terms across this set
- Unfortunately, Kemp (2019) approach only works well if the problem largely one dimensional, which is typically not the case for risk metrics such as VaR

Proposed simulation approach: tri-segmented Monte Carlo

- TSMC combines control variate methods with methods like Kemp (2019)
- Randomly draw **three** different simulations sets from underlying probability distribution
  - "Underlying" set, Z_u, used for estimating $\tilde{P}$ (or equivalent)
  - "Added" set, Z_a, used for estimating correction because $\tilde{P} \neq P$
  - "Extended" set, Z_e, used for estimating $\mathbb{E}(\tilde{P})$ or equivalent (c.f. Kemp (2019))
- Segment probability distribution space by the nearest point in Z_u ("Voroni cells") and choose $\tilde{P}$ to be best fit to nearest say n_{fit} points of Z_u to that cell
- Choose numbers in sets N_u, N_a and N_e so that $N_u \ll N_e$ and $N_a \ll N_e$. Evaluate P only $N_u + N_e$ times, but $\tilde{P}$ many more times

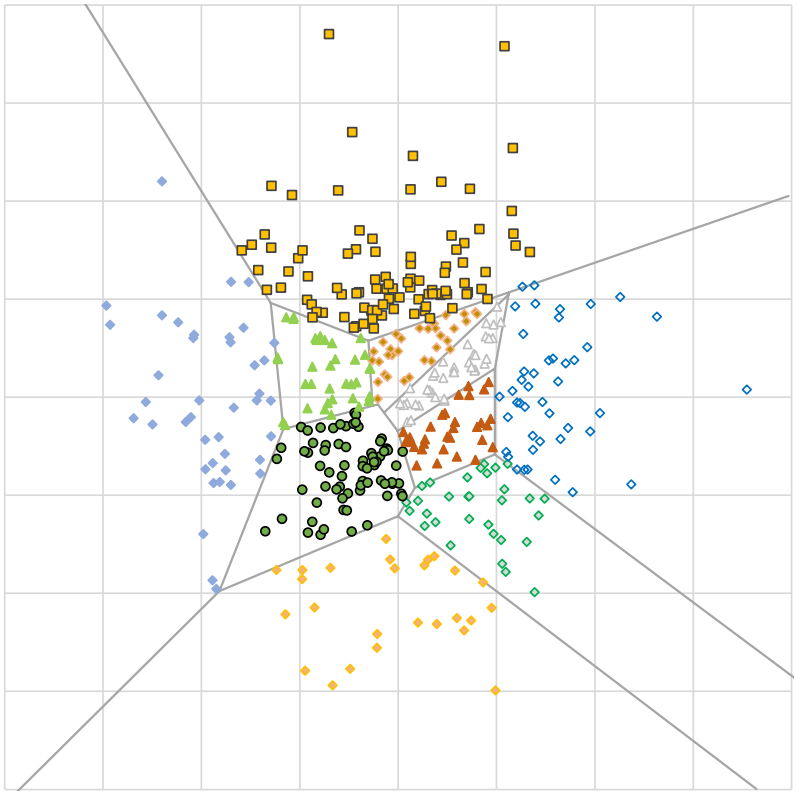
Step 1 (for portfolio valuation purposes)

- Identify an underlying simulation set Z_u and use it to create Voroni cells and in each cell an approximation $\tilde{P}$ to the true payoff P.
  - Voroni cells partition by the point in Z_u nearest to the relevant point in the space
  - Best fit $\tilde{P}$ using generalised linear regression and suitable basis functions, fitting to nearest other points in Z_u
  - Constrain $\tilde{P}$ so that $\tilde{P}(S_i^u) = P(S_i^u)$ for each S_i^u that is a member of Z_u (all other things being equal, S_i^u should be towards the middle of the relevant Voroni cell)



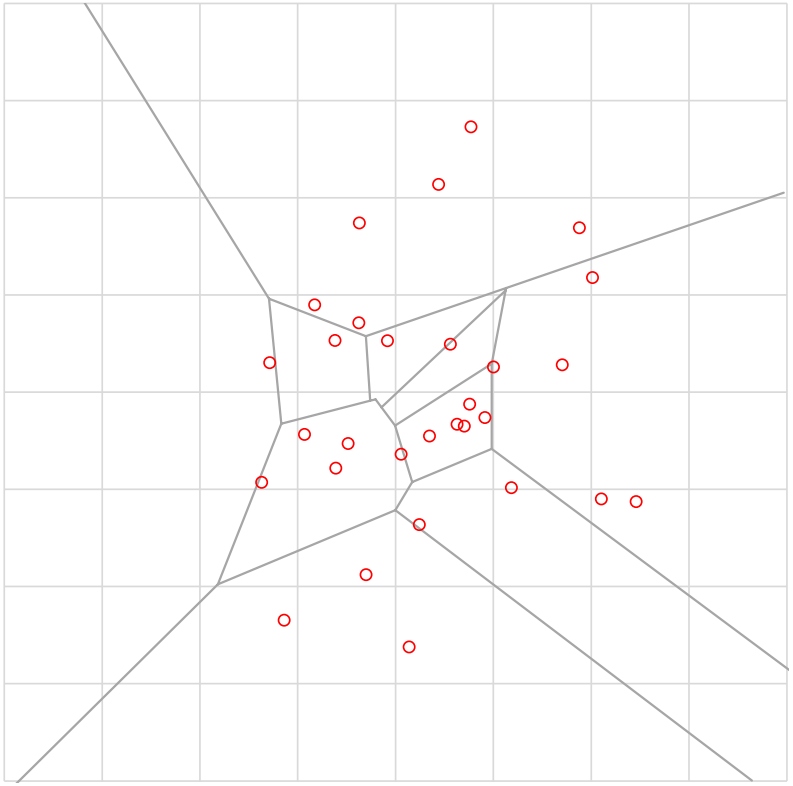
Step 2 (for portfolio valuation purposes)

- Identify an extended simulation set Z_e and estimate $\mathbb{E}(\tilde{P})$, the expected value of $\tilde{P}$, by averaging across these simulations



Step 3 (for portfolio valuation purposes)

- Identify an added simulation set Z_a and use it to estimate $\mathbb{E}\left(P - \tilde{P}\right)$, the expected value of $P - \tilde{P}$
- In practice probably do step 1 and 3 before step 2, to do all portfolio evaluations at the same time
- Potentially re-run for error estimation and/or bootstrap by randomly repartitioning $Z_u \cup Z_a$ between underlying and added simulation sets



Further comments

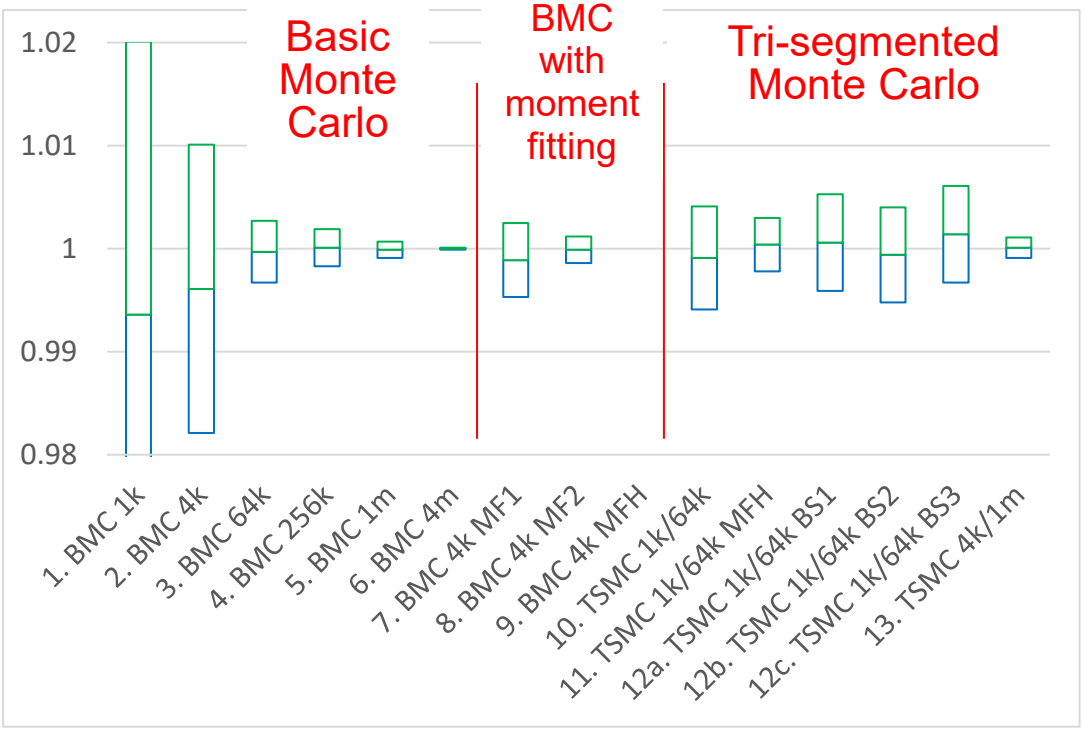
- For VaR:
  - Focus on losses, L, rather than present values, P. $\tilde{L}$ will be very close to L for points very near to a S_i^u but could on average be biased in relevant VaR tail
  - Could just assume a constant bias, but better seems to be to assume that there is a (e.g. linear) dependency between $L(S) - \tilde{L}(S)$ and $\tilde{L}(S)$. Estimate VaR from the relevant quantile of the adjusted $\tilde{L}(S)$ for the extended simulation set
- Simplest choice of $\tilde{P}$ is constant ($= P(S_i^u)$) within the cell defined by S_i^u
  - With this choice, subject to suitable regularity conditions, if N_u and N_a are large enough then, for large enough N_e, error will be less than for basic Monte Carlo
  - I.e. always optimal to use TSMC for large enough simulation sets and sufficiently time consuming to calculate payoffs (but unclear how large N_u and N_a need to be)
  - More sophisticated choices for $\tilde{P}$ (or $\tilde{L}$) likely to improve on constant in-cell $\tilde{P}$ (or $\tilde{L}$)

Illustrative exercise

- Test using a hypothetical moderate (i.e. 10) dimensional problem involving reasonably well diversified book capable of being valued analytically
  - 40 unit European-style options each on one of 2 distinct rolled up (i.e. non-dividend paying) indices following geometric Brownian motions (volatilities 10% pa and 20% pa), terms 1, 2, 3, 4 or 5 years, strikes 70%, 90%, 110% or 130%, interest rates assumed zero at outset and thereafter
- Distances between points in m-dimensional sample space (here $m = 10$) taken as Cartesian distance $|S_A - S_B| = \sqrt{\sum_{j=1}^{m} \left(S_{A,j} - S_{B,j}\right)^2}$
- $\tilde{P}$ chosen to involve best fit combinations of constant, linear, quadratic and "hockey stick" basis functions.
  - Simpler $\tilde{P}$ can be fitted to fewer (so closer) n_{fit} nearby points but otherwise likely to fit less well. Analogy with least-squares Monte Carlo. Including quadratic and "hockey stick" basis functions seems particularly helpful.

Estimated errors for portfolio values for various approaches

- Charts show average and ± 1 s.d. for 32 runs per approach (as fraction of true portfolio value):
  - Basic Monte Carlo shows $O(\varepsilon^{-2})$ dependency
  - Each instrument one-dimensional, can be valued well using high order moment fitting or Kemp (2019)
  - TSMC *m*/*n* provides error akin to BMC using *n* simulations but only evaluating portfolio payoff *m* times

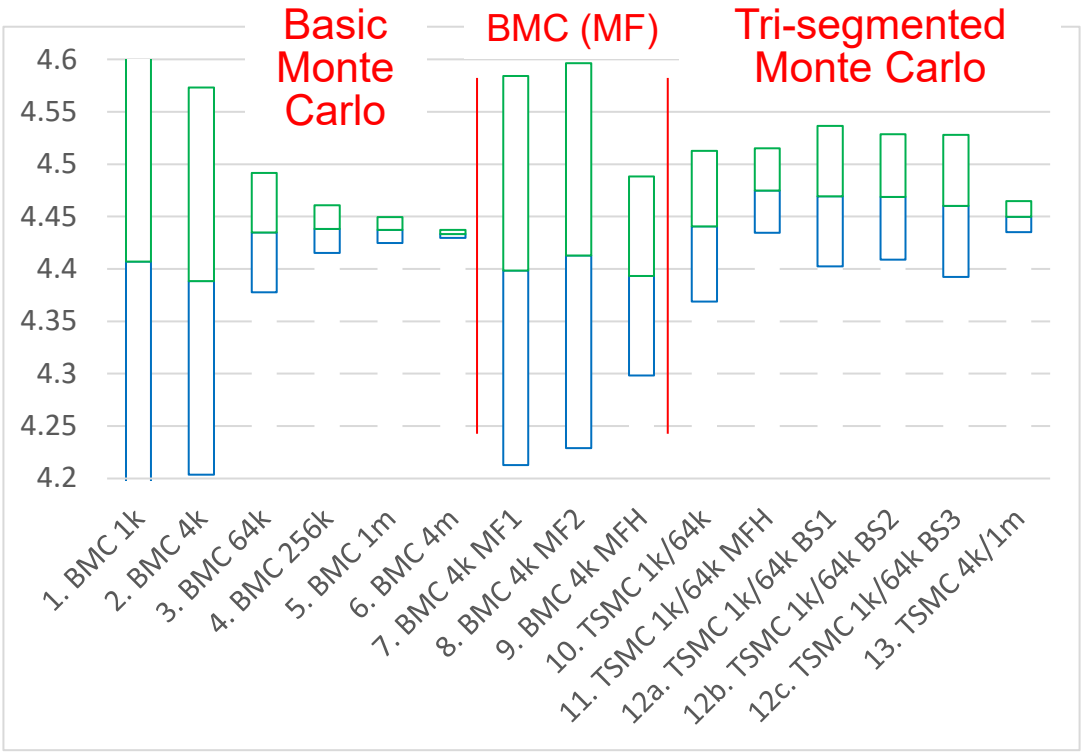


Source. Kemp (2022). Nomenclature:

- BMC n: Basic Monte Carlo with n simulations
- TMSC *m*/*n*: Tri-segmented Monte Carlo $m = N_u + N_a$, $n = N_e$ $(N_u = \frac{n}{4})$
- MF1, MF2, MFH: moment fitting approach if used where MF1 (fit observed to analytical means), MF2 (fit to analytical means + sds), MFH (fit to multiple lower moments); BS = bootstrapped from one of 3 singe selections of $Z_u \cup Z_a$

Corresponding results for 99.5% runoff loss quantiles

- C.f. “runoff” VaR (VaR to maturity)
  - Basic Monte Carlo still shows $O(\varepsilon^{-2})$ dependency
    - Moment fitting less effective (problem no longer 1-dimensional)
  - TSMC *m*/*n* error still like BMC *n*, but with a modest bias
- Bootstrapping TSMC seems to provide reasonable error estimates

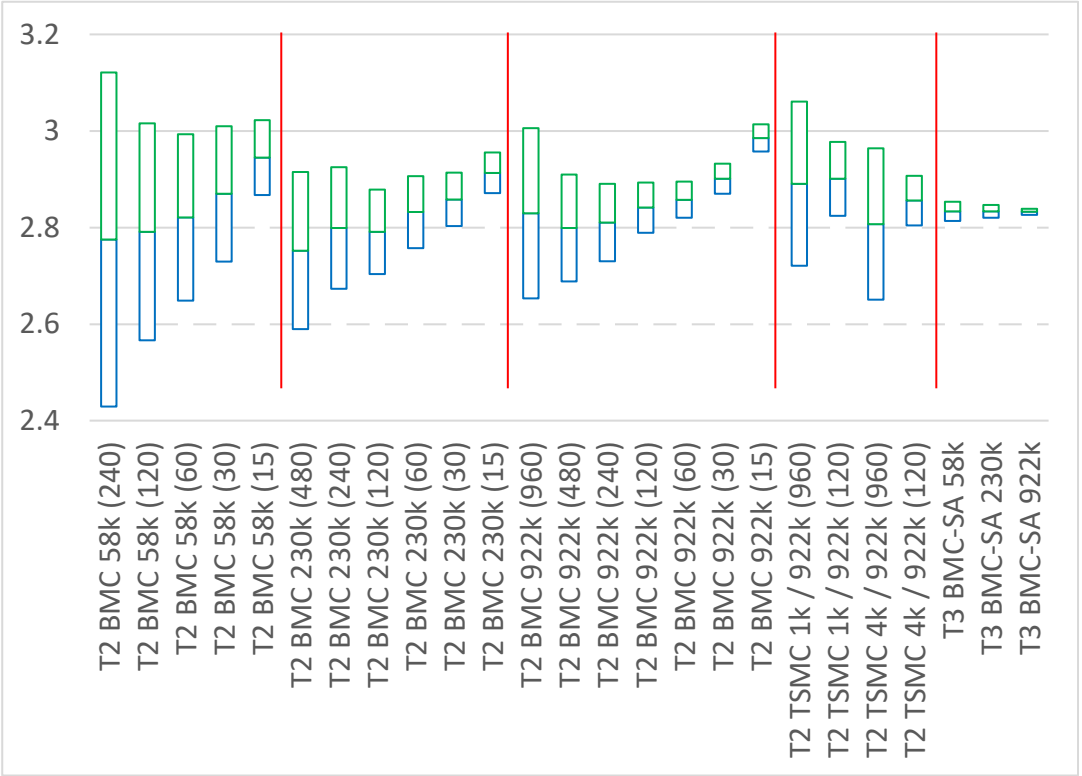


Source: Kemp (2022). Nomenclature:

- BMC n: Basic Monte Carlo with n simulations
- TMSC m/n: Tri-segmented Monte Carlo $m = N_u + N_a$, $n = N_e$ ($N_u = \frac{n}{4}$)
- MF1, MF2, MFH: moment fitting approach if used where MF1 (fit observed to analytical means), MF2 (fit to analytical means + sds), MFH (fit to multiple lower moments); BS = bootstrapped from one of 3 singe selections of $Z_u \cup Z_a$

Corresponding results for one-year 99.5% loss quantiles

- C.f. “1 year” VaR
  - Higher runtimes as in general a nested calculation (inner simulation to value portfolio at end year 1 conditional on outcome during year)
    - In this situation we can do the inner calculation analytically to estimate ‘true’ answer more precisely (see right hand end of chart)
  - TSMC *m*/*n* error still like BMC *n*
  - But again TSMC seems to exhibit a modest bias

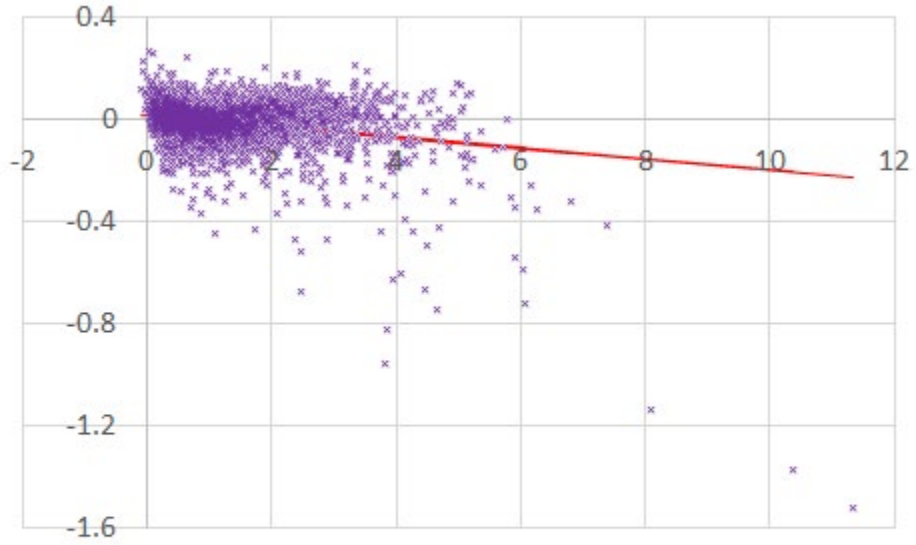


Source: Kemp (2022). Nomenclature:

- BMC *n* (*y*) and TSMC *m*/*n* (*y*): *n* overall simulations (for TSMC simulations within extended simulation set) with *y* inner simulations per outer simulation
- BMC-SA is “semi-analytical”, i.e. year 1 value derived analytically for each instrument, simulation applied only to year 1 evolution

VaR bias and other meta-parameters such as n_{fit}

- Analyse $L(S_j^a) - \tilde{L}(S_j^a)$ to infer likely behaviour of $L(S_i^e) - \tilde{L}(S_i^e)$
  - E.g. plot to visualise dependency on $\tilde{L}(S_j^a)$, to determine most appropriate form of VaR bias adjustment
  - E.g. quantify average and spread of $P(S_j^a) - \tilde{P}(S_j^a)$ or $L(S_j^a) - \tilde{L}(S_j^a)$ to place practical limits on VaR error, to estimate likely control variate variance ratio for different meta-parameters (to assist in selection of N_a and N_e relative to N_u) and to help select basis function types
- Z_u akin to a "training" set, Z_a akin to a "testing" set



Source: Kemp (2022). In chart, *x*-axis is $\tilde{L}(S_j^a)$ (as multiple of overall analytically derived portfolio value), *y*-axis is $L(S_j^a) - \tilde{L}(S_j^a)$. S_j^a are the points in Z_a.

Conclusions

- For sufficiently complex portfolios / nested calculations, tri-segmented Monte Carlo simulation always in the limit better runtime-wise than basic Monte Carlo
- If a complex portfolio behaved like the book illustrated here then significant runtime improvements might be achievable (64-fold or better)
- Finding the Voroni cell in which a given simulation sample lies can be time consuming but can be parallelised
- For actuaries, approach is akin to proxy modelling of the simulation set rather than of the liability profile
  - Proxying the simulation set may be easier to justify, e.g. by using analyses as per previous slide