Determining optimal multilevel Monte Carlo parameters with application to fault tolerance

Author(s):
Arbenz, Peter; Pauli, Stefan

Publication Date:
2014

Permanent Link:
https://doi.org/10.3929/ethz-a-010335876

Rights / License:
In Copyright - Non-Commercial Use Permitted
Determining Optimal Multilevel Monte Carlo Parameters with Application to Fault Tolerance

Stefan Pauli\textsuperscript{a,b,1}, Peter Arbenz\textsuperscript{a,*}

\textsuperscript{a}ETH Zürich, Computer Science Department, Universitätsstrasse 6, 8092 Zürich, Switzerland
\textsuperscript{b}ETH Zürich, Seminar for Applied Mathematics, Rämistrasse 101, 8092 Zurich, Switzerland

Abstract

The multilevel Monte Carlo (MLMC) method is characterized by a number of parameters, most notably the number of levels and the number of samples per level. We propose to determine these quantities by solving an integer optimization problem that minimizes the work or the error of the MLMC simulation. A branch-and-bound algorithm to solve these optimization problems is proposed and analyzed.

We investigate a fault tolerant MLMC method, in which samples affected by (hard) faults are discarded. As the failure rate increases more and more samples are lost. Thus, the average work to successfully complete a certain number of samples increases. The proposed optimization procedure can react on experienced faults and adapt the number of samples and levels accordingly. Numerical experiments demonstrate the effectiveness of the approach.

Keywords: Multilevel Monte Carlo, sample numbers, failure resilience

1. Introduction

In large scale simulations on emerging massively parallel compute platforms processor failures at runtime are inevitable [1] and occur, in fact, with increasing frequency as the number of processors gets larger. This motivated us to propose and implement a fault tolerant multilevel Monte Carlo (FT-MLMC) method [2, 3]. There it was assumed that the failure distribution of the computer is known. Unfortunately, only a few empirical statistical failure studies are available for modern computers [4, 5], hence realistic statistical failure models are rare. In this paper we present a FT-MLMC method which does not rely on the knowledge of the failure distribution.

In MLMC methods samples are computed on multiple levels. Samples on fine levels require much more compute resources (execution time, memory space, number of cores) than samples on coarse levels. By consequence, samples on fine levels are much more vulnerable to failures. In particular with high failure rates it may be very difficult to successfully complete samples on the finest level. Fortunately is is possible to modify the number of MLMC levels. It is, e.g., possible to drop some of the finest levels and spend the computing time on coarser levels instead and compute additional samples there. Depending on the problem, this may leads to a smaller MLMC error.
using the same computing time. A MLMC simulation with a high failure rate would benefit, as the failure-prone samples on the finest levels do not have to be computed.

Many MLMC applications benefit from a known optimal number of levels and the number of samples to be computed on each level. If the number of levels is fixed it is known how to compute optimal continuous sample numbers [6]. In [7] a method is described to find an optimal finest level. The approach is based on estimated convergence rates of level work, variance and discretion error. In our fault tolerant MLMC method with an unknown failure distribution a convergence of the level work may not be guaranteed, and hence this approach may not work. Additionally the approach does not operate with integer numbers of samples, which may lead to a suboptimal choice of the finest level.

In this paper we propose an integer optimization problem which is quickly solvable with the branch-and-bound algorithm. This provides a simple way to optimize both the number of levels as well as the integer optimal number of samples for each level.

The paper is organized as follows. In Section 2 we introduce MLMC methods. We present our algorithms to find the optimal finest level and the integer optimal number of samples in Section 3. In Sections 4 and 5 the algorithm is mathematically analyzed. In Section 6 a method is shown to find an optimal coarsest level. In Section 7 we describe our fault tolerant MLMC method using the findings of the previous sections. Finally, we show results in Section 8, which demonstrate the advantage of carefully choosing the finest level for FT-MLMC methods.

2. Multilevel Monte Carlo methods

The convergence rate of Monte Carlo (MC) methods, applied to estimate the expected solution of a partial differential equation with random input, is often suboptimal with respect to the computational work \( W \), whereas the MLMC method often reaches the optimal rate \( 1/\sqrt{W} \). In this section we provide a short introduction to MLMC methods. Complete descriptions are found, e.g., in [6, 8–10]. For an intuitive motivation see [3].

Both MC and MLMC methods can be used to approximate the expected value \( E[X(\omega)] \), \( \omega \in \Omega \), of a random variable \( X \) with sample space \( \Omega \). Usually an analytical \( X \) is unavailable, e.g., if \( X \) is the solution of a stochastic partial differential (PDE) equation. Instead we require a hierarchy of approximations \( X_\ell, \ell_0 \leq \ell \leq L \), from the least accurate \( X_{\ell_0} \) to the most accurate \( X_L \). In case of a stochastic PDE, the \( X \) maybe approximated by a finite volume solution \( X_\ell \) computed on a level-dependent mesh width \( h_\ell \), e.g., \( h_\ell = 2^{-\ell} \). The sequence of approximations has to converge in some Banach space \( B \),

\[
\|X - X_\ell\|_B \to 0 \quad \text{as} \quad \ell \to \infty.
\] (1)

As analytical \( X \) are not available we content ourselves with the expected value \( E[X_L] \). Using (1) and the linearity of the expectation we obtain

\[
\|E[X] - E[X_L]\|_B = \|E[X - X_L]\|_B \leq E[\|X - X_L\|_B] \to 0 \quad \text{as} \quad L \to \infty.
\]

MLMC exploits the provided hierarchy of approximations to express \( E[X_L] \) as a telescopic sum

\[
E[X_L] = \sum_{\ell=\ell_0+1}^{L} E[X_\ell - X_{\ell-1}] + E[X_{\ell_0}].
\] (2)
The latter is not computable due to the involved expectations \( \mathbb{E}[] \). Therefore, we approximate them by MC estimates

\[
\mathbb{E}[X_{\ell_0}] \approx \frac{1}{M_{\ell_0}} \sum_{m=1}^{M_{\ell_0}} X_{\ell_0}(\omega_m) =: E_{M_{\ell_0}}[X_{\ell_0}],
\]

and

\[
\mathbb{E}[X_\ell - X_{\ell-1}] \approx \frac{1}{M_\ell} \sum_{m=1}^{M_\ell} (X_\ell(\omega_m) - X_{\ell-1}(\omega_m)) =: E_{M_\ell}[X_\ell - X_{\ell-1}],
\]

using independent and identically distributed (i.i.d.) realizations \( \omega_m \). On each level \( \ell \) the MC estimate employs \( M_\ell \) samples. Replacing the expectations \( \mathbb{E}[] \) in (2) by MC estimates \( E_{M_\ell}[] \) leads to the MLMC estimate

\[
E[X_L] \approx \sum_{\ell=\ell_0+1}^{L} E_{M_\ell}[X_\ell - X_{\ell-1}] + E_{M_{\ell_0}}[X_{\ell_0}] =: E[X_L]
\]

(4)

Note that in (3) the same realization \( \omega_m \) is used for both approximations \( X_\ell(\omega_m) \) and \( X_{\ell-1}(\omega_m) \). The realizations \( \{\omega_m\}_{m=1}^{M_\ell} \) must be independent in each MC estimate \( \{E_{M_\ell}[]\}_{\ell=\ell_0}^{L} \) to ensure the statistical independence of the involved MC estimates.

An error bound for the MLMC estimate (4) is given by [8, 11]

\[
||\mathbb{E}[X] - E[X_L]||^2_{L^2(\Omega;Q)} \leq \frac{||\mathbb{E}[X - X_L]||^2_Q}{\text{discretization error}^2} + \frac{||\mathbb{E}[X_L] - E[X_L]||^2_{L^2(\Omega;Q)}}{\text{sampling error}^2}
\]

\[
\leq ||\mathbb{E}[X - X_L]||^2_Q + \sum_{\ell=\ell_0+1}^{L} \frac{1}{M_\ell} ||\mathbb{E}[X_\ell - X_{\ell-1}] - (X_\ell - X_{\ell-1})||^2_{L^2(\Omega;Q)}
\]

\[
+ \frac{1}{M_{\ell_0}} ||\mathbb{E}[X_{\ell_0}] - X_{\ell_0}||^2_{L^2(\Omega;Q)}
\]

\[
\approx \tau_L^2 + \sum_{\ell=\ell_0}^{L} \frac{\sigma_{\ell}^2}{M_\ell},
\]

(5)

where \( ||X||^2_{L^2(\Omega;Q)} := \mathbb{E}[||X||^2_Q] \) using a problem dependent norm \( Q \), for instance \( Q = L^2 \). The MLMC error decomposes into the discretization error \( \tau_L \), that only depends on the finest discretization level \( L \), and the sampling error \( \kappa \) that depends on the numbers of samples \( M_\ell \) and the level variance \( \sigma_{\ell}^2 \). The level variances in (5) are defined as follows:

\[
\sigma_{\ell}^2 = \begin{cases} 
(\sigma_{\ell}^2) = \frac{||\mathbb{E}[X_\ell - X_{\ell-1}] - (X_\ell - X_{\ell-1})||^2_{L^2(\Omega;Q)}}{\kappa^2}, & \text{if } \ell > \ell_0, \\
(\sigma_{\ell_0}^2) = \frac{||\mathbb{E}[X_{\ell_0}] - X_{\ell_0}||^2_{L^2(\Omega;Q)}}{\kappa^2}, & \text{if } \ell = \ell_0.
\end{cases}
\]

(6)

In most equations the level variance \( \sigma_{\ell}^2 \) will be used. Whenever we want to emphasize that the difference \( X_\ell - X_{\ell-1} \) of two solutions computed on consecutive discretization levels were used to
compute the level variance we will use \((\sigma^2_\Delta \ell)\) (for \(\ell > \ell_0\)). The level variance \((\sigma^2_I \ell_0)\) is used to emphasize that only one solution \(X_{\ell_0}\) computed on level \(\ell_0\) was used. A similar concept is used to define \(w_\ell\), the work per MLMC sample

\[
  w_\ell = \begin{cases} 
    w_\Delta^\ell = \text{Work to compute one sample } X_\ell - X_{\ell-1}, & \text{if } \ell > \ell_0, \\
    w_I^\ell = \text{Work to compute one sample } X_\ell, & \text{if } \ell = \ell_0.
  \end{cases}
\]

(7)

To use the MLMC method the discretization error \(\tau_L\), the level variance \(\sigma^2_\ell\) and the work \(w_\ell\) must be known, see equation (5). They are needed on the one hand to find good values for \(\ell_0\), \(L\) and \(M_\ell\) and on the other hand to bound the MLMC error (5). Two different strategies are common to determine \(\tau_L\), \(\sigma_\ell\) and \(w_\ell\). Either they can be bounded asymptotically [6, 8, 9] or they can be measured at runtime of the MLMC method [7, 11].

3. Discrete optimal numbers of samples

We propose to determine optimal values for \(L\) and \(\{M_\ell\}_{\ell=\ell_0}^L\) by solving an integer optimization problem. Below we state two related optimization problems. In the first we determine the parameters such that a result of a given accuracy is obtained with a minimal amount of work. In the second problem we determine the parameters such that the result is as accurate as possible given a certain work budget.

3.1. Constrained tolerance

We search for a set \(\{M_\ell\}_{\ell=\ell_0}^L\) that minimizes the work while the MLMC error (5) must not exceed a given tolerance \(\varepsilon\). For a fixed coarsest level \(\ell_0\), and known \(\tau_L\), \(\sigma_\ell\), and \(w_\ell\) the optimal integer sample numbers \(\{M_\ell\}\) and the optimal \(L\) can be found by minimizing the total MLMC work \(W_{\text{tot}} = \sum_{\ell=\ell_0}^L M_\ell w_\ell\) subject to the constraint that the MLMC error is not exceeding the predefined tolerance \(\varepsilon\). In some situations \(M_\ell^*\) samples may have been precomputed, e.g. to estimate \(w_\ell\). Then the optimization problem reads as

\[
  \begin{align*}
    \minimize_{\mathcal{N} \ni M_\ell \geq M_\ell^*} & \sum_{\ell=\ell_0}^L (M_\ell - M_\ell^*) w_\ell \\
    \text{subject to} & \varepsilon^2 \geq \kappa^2 + \tau^2_L, \\
    \text{constraint} & \kappa^2 := \sum_{\ell=\ell_0}^L \sigma^2_\ell / M_\ell
  \end{align*}
\]

(8)

3.2. Constrained work

Instead of achieving a predefined tolerance \(\varepsilon\) with minimal total work one may be interested in obtaining the smallest MLMC error without exceeding a predefined work budget \(W_{\text{tot}}\). The optimal integer numbers of samples \(\{M_\ell\}\) and the optimal \(L\) are then found by a modified version of the optimization problem (8), still with fixed coefficients \(\ell_0\), \(\sigma_\ell\), \(\tau_L\) and \(w_\ell\):

\[
  \begin{align*}
    \minimize_{\mathcal{N} \ni M_\ell \geq M_\ell^*} & \kappa^2 + \tau^2_L, \\
    \text{subject to} & W_{\text{tot}} \geq \sum_{\ell=\ell_0}^L (M_\ell - M_\ell^*) w_\ell \\
    \text{constraint} & \kappa^2 := \sum_{\ell=\ell_0}^L \sigma^2_\ell / M_\ell
  \end{align*}
\]

(9)
3.3. A branch-and-bound algorithm

To solve the optimization problems and to find optimal discrete MLMC parameters we use a starting level \( L_{\text{start}} \) as an input, which is an upper bound on the optimal finest level \( L \). A valid starting level is for instance found if the total work budget is smaller than what it would cost to compute one single sample on level \( L_{\text{start}} \), \( W_{\text{tot}} < w_{L_{\text{start}}} \).

On each level \( \ell_{0} \leq \ell \leq L_{\text{start}} \) a sample number \( M_{\ell} \) has to be determined. As soon as the sample numbers \( \{M_{\ell}\} \) are known the optimal finest level \( L \) is the finest level on which a sample is computed

\[
L = \max_{\ell} (\ell \mid M_{\ell} > 0). \tag{10}
\]

Note that finitely many configurations \( \{M_{\ell}\}_{\ell=L_{\text{start}}}^{L_{\text{start}}-1} \) exist, as the maximal number of samples per level is bounded by, e.g., \( M_{\ell}^{\text{max}} \leq W_{\text{tot}}/w_{\ell} \). We organize the possible configurations \( \{M_{\ell}\}_{\ell=L_{\text{start}}}^{L_{\text{start}}-1} \) in a tree, see Figure 1. The root of the tree corresponds to level \( L_{\text{start}}+1 \) where no samples are computed. The leaves correspond to the coarsest level \( \ell_{0} \). Each configuration \( \{M_{\ell}\} \) finds its representation in a path from the root to a leaf.

A naive algorithm may test each possible configuration \( \{M_{\ell}\}_{\ell=L_{\text{start}}}^{L_{\text{start}}-1} \) by executing some traversal of the tree. If a configuration satisfies the constraints, then the objective function is evaluated. Among all admissible configurations those that minimize the objective function are optimal.

This algorithm can be accelerated by a branch-and-bound approach [12]. The procedure rests on the fact that the solution of the problems (8) and (9) can be given explicitly if the sample numbers are admitted to be nonnegative reals \( (M_{\ell} \in \mathbb{R}_{+}) \). So, at a node on level \( \ell \) of the tree, characterized by the path \( (M_{L_{\text{start}}}, \ldots, M_{\ell}) \), we can cheaply get a lower bound of the objective function by setting the undetermined numbers \( M_{\ell-1}, \ldots, M_{\ell_{0}} \) equal to the solution \( \tilde{M}_{\ell-1}, \ldots, \tilde{M}_{\ell_{0}} \) of the relaxed (continuous) optimization problem. If the lower bound is bigger than the actual minimal value then we can prune the subtree starting at the node \( (M_{L_{\text{start}}}, \ldots, M_{\ell}) \).

We can further improve the procedure by starting on level \( \ell \) of the tree with a value \( \overline{M}_{\ell} \) that is close to \( \tilde{M}_{\ell} \) which in turn is obtained by solving a relaxed problem at level \( \ell + 1 \). The search is extended in both directions starting at \( \overline{M}_{\ell} \). In section 4 we show that the lower bound for the objective function is in fact a convex function in \( M_{\ell} \). Therefore, if the lower bound is bigger than the minimal value found so far the corresponding subtree can be pruned together with all subtrees that originate from nodes on level \( \ell \) that are further away from \( \overline{M}_{\ell} \).
Algorithm 3.1 Find optimal values for parameters $L, M_{\ell_0}, \ldots, M_L$.

function $\text{intOpt}(L, k, M_{k+1}, \ldots, M_L)$

Given the numbers of precomputed samples $\{M^*_\ell\}_{\ell=0}^{L_{\text{start}}}$, a coarsest level $\ell_0$ and the parameters $L_{\text{start}}, \tau_L, \sigma_\ell, w_\ell$. Find an optimal finest level $L$ and the corresponding optimal integer numbers of samples $M_\ell$, $\ell_0 \leq \ell \leq L$:

Use global variables for the upper bound of the objective function $U_{\text{OF}}$ and the coarsest level $\ell_0$. Initialize the variable $U_{\text{OF}} = \infty$. To solve the integer optimization problem execute the following recursive algorithm $\text{intOpt}(L_{\text{start}}, L_{\text{start}})$ with a large enough starting level $L_{\text{start}}$.

1: if $k = \ell_0$ then
2: Compute optimal integer value for $M_{\ell_0}$
3: if $U_{\text{OF}} > \text{OF}(M_{\ell_0}, \ldots, M_L)$ then
4: $U_{\text{OF}} = \text{OF}(M_{\ell_0}, \ldots, M_L)$
5: \{$M_{\ell_0}, \ldots, M_L$ and $L$ are so far the best integer parameters.\}
6: else
7: Compute the optimal continuous value for $\tilde{M}_k$.
8: iterate over all $M_k \in \mathbb{N}$, $M_k \geq M^*_k$, ordered by distance to $\tilde{M}_k$, with $L_{\text{OF}}(M_k, \ldots, M_L) < U_{\text{OF}}$
9: $\text{intOpt}(L, k-1, M_k, \ldots, M_L)$
10: end iterate
11: if $M_k = 0$ and $L = k$ then
12: $\text{intOpt}(k-1, k-1)$
13: end if
14: end if
15: end if

The pseudo code presented in Algorithm 3.1 implements the above branch-and-bound algorithm. Details on the lower bound of the objective function $L_{\text{OF}}(M_k, \ldots, M_L)$ are provided in Sections 4 and 5.

3.4. Optimizing with unknown parameters

Algorithm 3.1 finds optimal $\{M_\ell\}$ and $L$ based on a fix $\ell_0$ and known parameters $w_\ell, \sigma_\ell$ and $\tau_L$. In applications some or all of these parameters may not be known a priori but can be approximated during the MLMC execution [11, 13, 14]. We extend an algorithm of [15], that simultaneously computes MLMC samples and estimates the unknown parameters, with the ability to determine optimal values for $\{M_\ell\}$ and $L$.

The algorithm starts by computing a moderate number of samples on a few levels, then uses them to measure the unknown parameters, and, finally, adds new levels step by step and computes more samples. In this algorithm we fix two levels $\ell_0$ and $\ell_1$ with $\ell_0 \leq \ell_1 \leq L$. We start by computing $K$ samples (e.g. $K = 10$) on levels $\ell_0$ to $\ell_1$ and use these samples to measure $w_\ell, \sigma_\ell$ and $\tau_\ell$ for levels $\ell_0 \leq \ell \leq \ell_1$. Then, based on the measurements, we estimate $w_{\ell_1+1}, \sigma_{\ell_1+1}$ and $\tau_{\ell_1+1}$. These estimates are usually based on the theory about the behavior of the parameters. Using these parameters, we solve an optimization problem (apply Algorithm 3.1 with $L_{\text{start}} = \ell_1 + 1$), which provides a set of sample numbers $\{M_\ell\}_{\ell=\ell_1}^{L_{\text{start}}+1}$. If a new level is added ($M_{\ell_1+1} > 0$), then at first at most $K$ samples are computed on this level ($\min(K, M_{\ell_1+1})$), i.e. only a fraction of the proposed sample number. On all other levels we compute additional samples until the same fraction of $M_\ell$
is compute. Then we increase $\ell_1$ by one. If no level is added ($M_{\ell_1+1} = 0$) then we compute the additional samples required to reach the proposed samples on all levels. In both cases we iteratively loop in the algorithm, and adjust the measurements and estimates of $w_\ell$, $\sigma_\ell$ and $\tau_\ell$, $\ell_0 \leq \ell \leq \ell_1 + 1$, with the newly computed samples. The algorithm terminates, once no additional samples remain to be computed.

As suggested in [15] we only compute 70% of the samples proposed by the algorithm. This prevents an overshoot of the computed samples as a result of the potentially inaccurately measured parameters $w_\ell$, $\sigma_\ell$, and $\tau_\ell$.

4. Definition and analysis of a lower bound for the problem with constrained tolerance

In this section we present a lower bound $L_{\text{OF}}$ for the objective function $W_{\text{tot}}$ in (8) where we want to find a result of prescribed accuracy with as little work as possible. The key idea is to extend the range of admissible values of the $\{M_\ell\}$ to the non-negative real numbers $\mathbb{R}_+$. This ‘continuous’ optimization problem evidently has a smaller minimum than the corresponding discrete problem. The ‘continuous’ minimum can be computed cheaply and serves as a lower bound for the ‘discrete’ minimum.

We use this idea in the branch-and-bound algorithm 3.1. On level $k$ of the search tree we require a lower bound $L_{\text{OF}}(M_k, \ldots, M_L)$ of the objective function $W_{\text{tot}}(M_k, \ldots, M_L)$ where the sample numbers $M_k, \ldots, M_L \in \mathbb{N}$ are prescribed and the $M_{\ell_0}, \ldots, M_{k-1} \in \mathbb{R}_+$ are to be determined.

If the lower bound is bigger than the best configuration that we found so far, we can prune the whole subtree starting at $(M_k, \ldots, M_L)$. We further show that the lower bound $L_{\text{OF}}(M_k, \ldots, M_L)$ as a function of $M_k$ is convex. This means that we have to evaluate $L_{\text{OF}}(M_k, \ldots, M_L)$ only for a few values $M_k \in \mathbb{N}$ in the vicinity of the minimum.

4.1. A relaxed problem

We modify the minimization problem (8) by admitting continuous sample numbers ($M_\ell \in \mathbb{R}_+$), and by using a fixed finest level $L$. If no samples are precomputed ($M_\ell^* = 0$) a straightforward application of the method of Lagrange multipliers yields the solution of the continuous optimization problem, 

$$M_\ell = \frac{1}{\kappa^2} \sigma_\ell \sqrt{w_\ell} \sum_{i=\ell_0}^L \sigma_i \sqrt{w_i}, \quad \kappa^2 = \epsilon^2 - \tau^2_L.$$ 

(11)

If $M_\ell^* > 0$ for some $\ell$ we apply the greedy Algorithm 4.1 to adapt the $M_\ell$.

The algorithm scales linearly in the number of levels if $M_\ell^* = 0$, otherwise, in the worst case, quadratically. However, also large MLMC simulations only use a few levels, and hence the computation of continuous sample numbers is fast.

4.2. A lower bound

In Algorithm 3.1 we require a lower bound $L_{\text{OF}}(M_k, \ldots, M_L)$ of the objective function. There the numbers of samples $M_k, \ldots, M_L \in \mathbb{N}$ are fixed whereas the $M_{\ell_0}, \ldots, M_{k-1} \in \mathbb{R}_+$ are to be determined. This allows to split the constraint of the optimization problem (8) into a known and an unknown portion 

$$\frac{\epsilon^2}{\text{known}} \geq \sum_{i=\ell_0}^{\ell-1} \frac{\sigma^2_i}{M_i} + \sum_{i=k}^{L} \frac{\sigma^2_i}{M_i} + \tau^2_L.$$ 

(12)
Algorithm 4.1 Find continuous sample numbers $M_\ell$ that lead to a small total work $W_{tot}$

Given the number of precomputed samples $M_\ell^* \geq 0$ and the parameters $\ell_0$, $L$, $\kappa$, $\tau_L$, $\sigma_\ell$, and $w_\ell$. Find sample numbers $M_\ell \in \mathbb{R}_+$ that give a small total work $W_{tot}$.

1: Define the set $\mathcal{L} := \{\ell_0, \ldots, L\}$. Set $\kappa^2 = \epsilon^2 - \tau_L^2$.
2: for all $\ell \in \mathcal{L}$ do
3:     Set $M_\ell = \frac{1}{\kappa^2} \frac{\sigma_\ell}{\sqrt{w_\ell}} \sum_{i \in \mathcal{L}} \sigma_i \sqrt{w_i}$.
4: end for
5: for all $\ell \in \mathcal{L}$ with $M_\ell < M_\ell^*$ do
6:     remove $\ell$ from $\mathcal{L}$, set $M_\ell = M_\ell^*$ and $\kappa^2 = \kappa^2 - \frac{\sigma_\ell^2}{M_\ell^*}$.
7: end for
8: if no $\ell$ has been removed from $\mathcal{L}$ then
9:     quit
10: else
11:     go to 2:
12: end if

Rewriting the above formula permits to determine the maximal sampling error $\tilde{\kappa}$ on the “unknown” levels $\ell_0, \ldots, k - 1$ such that the constraint still holds

$$\tilde{\kappa}^2 = \kappa^2 - \frac{\sigma_k^2}{M_k} = \sum_{\ell=\ell_0}^{k-1} \frac{\sigma_\ell^2}{M_\ell} \leq \epsilon^2 - \sum_{\ell=k}^{L} \frac{\sigma_\ell^2}{M_\ell} - \tau_L^2.$$  

Using Algorithm 4.1, with $\kappa = \tilde{\kappa}$ and $L = k - 1$, we compute optimal continuous sample numbers $\tilde{M}_\ell \geq M_\ell^*$, $\ell_0 \leq \ell \leq k - 1$. In case no samples are precomputed ($M_\ell^* = 0$) this results in

$$\tilde{M}_\ell = \frac{1}{\tilde{\kappa}^2} \frac{\sigma_\ell}{\sqrt{w_\ell}} \sum_{i=\ell_0}^{k-1} \sigma_i \sqrt{w_i},$$  

where it is easily observed, that the more we invest on level $k$ the less samples are required on the other levels $\ell_0 \leq \ell < k$ (note that $\frac{dM_\ell}{dM_k} < 0$). Using the continuous optimal $\tilde{M}_{\ell_0}, \ldots, \tilde{M}_{k-1}$ in the objective function (8) the lower bound for the total work becomes

$$L \sum_{\ell=\ell_0}^{k-1} \frac{(\tilde{M}_\ell - M_\ell^*) w_\ell}{\tilde{M}_\ell} + \sum_{\ell=k}^{L} (M_\ell - M_\ell^*) w_\ell.$$  

4.3. Convexity of the lower bound

We now prove the convexity of the lower bound as a function of $M_k$. We first examine the special case where no samples are precomputed ($M_\ell^* = 0$ for all $\ell$). In this case the lower bound can be given analytically by inserting the optimal continuous sample numbers of (11) into the lower bound (14),

$$L \sum_{\ell=\ell_0}^{k-1} \frac{(\tilde{M}_\ell - M_\ell^*) w_\ell}{\tilde{M}_\ell} + \sum_{\ell=k}^{L} M_\ell w_\ell.$$  

Here, we used $\hat{\kappa}^2 = \hat{\kappa}^2 - \sigma_k^2/M_k$, where $\hat{\kappa}^2 = \sum_{\ell=\ell_0}^k \sigma_{\ell}^2/M_\ell$ denotes the sampling error on levels $\ell_0$ to $k$.

The first derivative of $L.W_{\text{tot}}$ with respect to $M_k$ reveals its minima and maxima. The equation

$$\frac{d}{dM_k} L.W_{\text{tot}}(M_k, \ldots, M_L) = w_k - \frac{\sigma_k^2 \left( \sum_{\ell=\ell_0}^{k-1} \sigma_{\ell} \sqrt{w_{\ell}} \right)^2}{(\sigma_k^2 - \hat{\kappa}^2 M_k)^2} = 0$$

has the two solutions

$$M_k = \frac{\sigma_k^2}{\hat{\kappa}^2} \pm \frac{\sigma_k (\sum_{\ell=\ell_0}^{k-1} \sigma_{\ell} \sqrt{w_{\ell}})}{\hat{\kappa}^2 \sqrt{w_k}}.$$  

As all involved variables are positive, only the one with positive sign satisfies $M_k > \sigma_k^2/\hat{\kappa}^2$. The other solution is infeasible ($\hat{\kappa}^2 = \sum_{\ell=\ell_0}^k \sigma_{\ell}^2/M_\ell < \sigma_k^2/M_k$).

The second derivative

$$\frac{d^2}{dM_k^2} L.W_{\text{tot}}(M_k, \ldots, M_L) = \frac{2 \hat{\kappa}^2 \sigma_k^2 \left( \sum_{\ell=\ell_0}^{k-1} \sigma_{\ell} \sqrt{w_{\ell}} \right)^2}{(\hat{\kappa}^2 M_k - \sigma_k^2)^3}$$

is positive for $M_k > \sigma_k^2/\hat{\kappa}^2$. Hence $L.W_{\text{tot}}$ is convex and has a unique minimum in the range of interest.

Now we show that the bound remains convex if there are precomputed samples, $M^*_\ell > 0$ for some $\ell$. Note that without precomputed samples the continuous sample numbers $\{M_\ell\}$ in (13) can only decrease with increasing $M_k$. With precomputed samples, the decrease is limited by $M^*_\ell$. This implies that for a level with $M^*_\ell > 0$ there is a certain transition point $M^*_k$ where the case $\hat{\kappa} = M^*_k$ switches to the case $M_\ell = M^*_\ell$. Let $\mathcal{Z} = \{ \ell < k | M^*_k \leq M^*_k \}$ and let $\mathcal{L} = \{\ell_0, \ldots, k-1\} \setminus \mathcal{Z}$ denote the complementary set. Just below the transition point $M^*_k$ of level $j$ the sample numbers on the levels $\ell \in \mathcal{Z}$ are fixed, $M_\ell = M^*_\ell$. At the transition point $M^*_k$ the sets change to $\mathcal{Z} = \mathcal{Z} \cup \{j\}$ and $\mathcal{L}' = \mathcal{L} \setminus \{j\}$.

Below (at least in the immediate vicinity of) $M^*_k$ the lower bound is given by

$$L.W_{\text{tot}, \mathcal{L}}(\cdot) = \frac{1}{\hat{\kappa}^2_{\mathcal{L}}} \left( \sum_{\ell \in \mathcal{L}} \sigma_{\ell} \sqrt{w_{\ell}} \right)^2 + \sum_{\ell=k}^{L} (M_\ell - M^*_\ell) w_\ell - \sum_{\ell \in \mathcal{L}} M^*_\ell w_\ell$$

and above $M^*_k$ by

$$L.W_{\text{tot}, \mathcal{L}'}(\cdot) = \frac{1}{\hat{\kappa}^2_{\mathcal{L}'}} \left( \sum_{\ell \in \mathcal{L}'} \sigma_{\ell} \sqrt{w_{\ell}} \right)^2 + \sum_{\ell=k}^{L} (M_\ell - M^*_\ell) w_\ell - \sum_{\ell \in \mathcal{L}'} M^*_\ell w_\ell,$$

with

$$\hat{\kappa}^2_{\mathcal{L}} = \hat{\kappa}^2 - \sum_{\ell \in \mathcal{Z}} \frac{\sigma_{\ell}^2}{M^*_\ell}, \quad \hat{\kappa}^2_{\mathcal{L}'} = \hat{\kappa}^2 - \frac{\sigma_{\ell}^2}{M^*_\ell}.$$  

$\hat{\kappa}$ corresponds to the allowed sampling error on the levels $\ell_0 \leq \ell \leq k$. $\hat{\kappa}_{\mathcal{L}}$ excludes the levels $\ell \in \mathcal{Z}$ where $M_\ell = M^*_\ell$. 

9
The lower bound $L_{\text{tot},\mathcal{L}}$ is derived from (14) where not only the sample numbers on levels $k \leq \ell \leq L$ are known, but also $M_\ell = M^*_\ell$ on $\ell \in \mathbb{Z}$. The optimal continuous sample numbers on the levels $\ell \in \mathcal{L}$ are obtained by Algorithm 4.1 using the predefined set $\mathcal{L}$, $\kappa = \hat{\kappa}_L$ and $L = k - 1$.

Above $M^j_\ell$ the first and second derivatives of $L_{\text{tot},\mathcal{L}'}$ are given by

\[
\frac{d}{dM_k} L_{\text{tot},\mathcal{L}'}(M_k, \ldots, M_L) = w_k - \frac{\sigma^2_k \left( \sum_{\ell \in \mathcal{L}'} \sigma_\ell \sqrt{w_\ell} \right)^2}{(\sigma^2_k - \hat{\kappa}^2_k M_k)^2},
\]

\[
\frac{d^2}{d^2 M_k} L_{\text{tot},\mathcal{L}'}(M_k, \ldots, M_L) = 2 \frac{\hat{\kappa}_L^2 \sigma^2_k \left( \sum_{\ell \in \mathcal{L}'} \sigma_\ell \sqrt{w_\ell} \right)^2}{(\hat{\kappa}_L^2 M_k - \sigma^2_k)^3}.
\]

The formulas for $L_{\text{tot},\mathcal{L}}$ are similar.

We now determine the transition point $M^*_j$, i.e., the point where $M_j$ becomes $M^*_j$ in $L_{\text{tot},\mathcal{L}}$. Adapting (12) we get

\[
M^*_j = M_j = \frac{1}{\hat{\kappa}_L^2} \frac{\sigma_j}{M_j} \frac{\sigma_j}{M_j} \frac{\sigma_j}{\sqrt{w_j}} \frac{\sigma_j}{\sqrt{w_j}} \sum_{\ell \in \mathcal{L}'} \sigma_\ell \sqrt{w_\ell},
\]

from which we obtain

\[
M^*_j = \frac{\sigma^2_k}{\hat{\kappa}_L^2} \frac{\sigma_j}{M_j} \frac{\sigma_j}{\sqrt{w_j}} \frac{\sigma_j}{\sqrt{w_j}} \sum_{\ell \in \mathcal{L}'} \sigma_\ell \sqrt{w_\ell}.
\]

Note that

\[
M^*_j = \frac{\sigma^2_k}{\hat{\kappa}_L^2} \frac{\sigma_j}{M_j} \frac{\sigma_j}{\sqrt{w_j}} \frac{\sigma_j}{\sqrt{w_j}} \sum_{\ell \in \mathcal{L}'} \sigma_\ell \sqrt{w_\ell} > \frac{\sigma^2_k}{(\hat{\kappa}_L^2 - \sigma^2_j)^2} = \frac{\sigma^2_k}{\hat{\kappa}_L^2}.
\]

Hence the first and second derivatives are well defined. The continuity of $\frac{d}{dM_k} L_{\text{tot}}$ can easily be verified by solving $\frac{d}{dM_k} L_{\text{tot},\mathcal{L}'} = \frac{d}{dM_k} L_{\text{tot},\mathcal{L}'}$ for $M_k$ and showing that (15) holds.

We showed that the first derivative of $L_{\text{tot}}$ is continuous, that the second derivative is positive for $M_k > \sigma^2_k/\hat{\kappa}^2$ and the lower bound $L_{\text{tot}}$ is convex with respect to $M_k$. Hence, for the continuous case, a unique minimum $M_k > \sigma^2_k/\hat{\kappa}^2$ exists.

5. Definition and analysis of a lower bound for the problem with constrained work

We modify the minimization problem (9) by admitting continuous sample numbers ($M_\ell \in \mathbb{R}_+$), and by using a fixed finest level $L$. If no samples are precomputed ($M^*_\ell = 0$) a straightforward application of the method of Lagrange multipliers yields the solution of the continuous optimization problem,

\[
M_\ell = W_{\text{tot}} \frac{\sigma_\ell}{\sqrt{w_\ell}} \left( \frac{1}{\sum_{i=L_0}^L \sigma_i \sqrt{w_i}} \right)^{-1}.
\]

If $M^*_\ell > 0$ for some $\ell$ we apply the greedy Algorithm 5.1.
Algorithm 5.1 Find continuous sample numbers $M_{\ell}$ that entail a small error $\kappa^2 + \tau_L^2$

Given the number of precomputed samples $M^*_\ell \geq 0$ and the parameters $\ell_0$, $L$, $W_{\text{tot}}$, $\tau_L$, $\sigma_\ell$, and $w_\ell$. Find sample numbers $M_{\ell} \in \mathbb{R}_+$ with a small error $\kappa^2 + \tau_L^2$.

1: Define the set $\mathcal{L} := \{\ell_0, \ldots, L\}$.

2: for all $\ell \in \mathcal{L}$ do

3: Set

$$M_{\ell} = \left( W_{\text{tot}} + \sum_{k \in \mathcal{L}} M^*_k w_k \right) \frac{\sigma_\ell}{\sqrt{w_\ell}} \left( \sum_{k \in \mathcal{L}} \sigma_k \sqrt{w_k} \right)^{-1}.$$ 

4: end for

5: for all $\ell \in \mathcal{L}$ with $M_{\ell} < M^*_\ell$ do

6: remove $\ell$ from $\mathcal{L}$, set $M_{\ell} = M^*_\ell$ and $W_{\text{tot}} = W_{\text{tot}} - M_{\ell} w_\ell$.

7: end for

8: if no $\ell$ has been removed from $\mathcal{L}$ then

9: quit

10: else

11: go to 2;

12: end if

5.1. A lower bound

We follow the main ideas of Section 4.2 to derive a pruning rule, this time for the constrained work, by computing continuous optimal $\tilde{M}_{\ell_0}, \ldots, \tilde{M}_{k-1}$ and by bounding the objective function. The constraint is split into a known and an unknown portion, but this time the constraint of the optimization problem (9) is used

$$W_{\text{tot}} \geq \sum_{\ell = \ell_0}^{k-1} (M_{\ell} - M^*_\ell) w_\ell + \sum_{\ell = k}^{L} (M_{\ell} - M^*_\ell) w_\ell.$$ 

Hence we can compute the work $\tilde{W}_{\text{tot}}$ which can be spent on the “unknown” levels $\ell_0, \ldots, k - 1$ such that the total work does not exceed the given work budget. Then the optimized continuous number of samples $\tilde{M}_{\ell_0}, \ldots, \tilde{M}_{k-1}$ can be computed as indicated in Section 3.2 with the total work budget of $\tilde{W}_{\text{tot}}$, and the finest unknown level $k - 1$. Then we split the objective function into a known and an unknown portion

$$\varepsilon^2 \geq \sum_{\ell = \ell_0}^{k-1} \frac{\sigma_\ell^2}{M_{\ell}} + \sum_{\ell = k}^{L} \frac{\sigma_\ell^2}{M_{\ell}} + \tau_L^2.$$ 

In the known portion we will use the know integer $M_k, \ldots, M_L$ and in the unknown portion we use the continuous optimal $\tilde{M}_{\ell_0}, \ldots, \tilde{M}_{k-1}$. This defines a lower bound of the objective function by

$$L \varepsilon^2(M_k, \ldots, M_L) = \sum_{\ell = \ell_0}^{k-1} \frac{\sigma_\ell^2}{M_{\ell}} + \sum_{\ell = k}^{L} \frac{\sigma_\ell^2}{M_{\ell}} + \tau_L^2.$$ 

11
5.2. Convexity of the lower bound

We define the sets $\mathcal{L}$ and $\mathcal{L}$ as in Section 4.3, where the convexity of the lower bound is proven for the constrained work. The lower bound

$$L_\varepsilon^2(\cdot) = \frac{1}{W_{\text{tot},\mathcal{L}}} \left( \sum_{\ell \in \mathcal{L}} \sigma_\ell \sqrt{w_\ell} \right)^2 + \sum_{\ell=k}^{L} \frac{\sigma_\ell^2}{M_\ell^2} + \tau_\ell^2 + \sum_{\ell \in \mathcal{L}} \sigma_\ell \sqrt{w_\ell} + \tau_{\ell + 1}^2 \sum_{\ell \in \mathcal{L}} \sigma_\ell \sqrt{w_\ell}$$

with

$$W_{\text{tot},\mathcal{L}} = W_{\text{tot}} - \sum_{\ell=k}^{L} (M_\ell - M_\ell^*) w_\ell + \sum_{\ell \in \mathcal{L}} M_\ell^* w_\ell$$

and the constraint

$$\sum_{\ell=k}^{L} (M_\ell - M_\ell^*) w_\ell \leq W_{\text{tot}}$$

however differs from Section 4.3. The constraint is physically meaningful, as the work required to compute the samples on levels $k \leq \ell \leq L$ must not exceed the total work. The proof of the convexity of the lower bound with respect to $M_k$ requires similar steps as in Section 4.3. We do not show them here.

6. Coarsest level $\ell_0$

In the previous optimization problems the coarsest level $\ell_0$ has been prescribed. In this section we provide an algorithm to find an optimal coarsest level $\ell_0$. The idea behind this algorithm is to describe the total MLMC work as a function of $\ell_0$. Then start with the coarsest possible level $\ell_0$ (e.g. a single cell in the finite element method) and increase $\ell_0$ until the total MLMC work increases.

We compute the total MLMC work to achieve a certain tolerance $\varepsilon$, in order to find the $\ell_0$ resulting in the smallest total work. We assume that $L$ is large enough, such that the discretization error does not exceed the tolerance $\varepsilon < \tau_L$. The total work is computed using the optimal $M_\ell$ from (11). We write the total work as a function of the coarsest level $\ell_0$:

$$W_{\text{tot}}(\ell_0) = \sum_{\ell=\ell_0}^{L} M_\ell w_\ell = \frac{1}{\varepsilon^2 - \tau_L^2} \sum_{k=\ell_0}^{L} \sigma_k \sqrt{w_k} \cdot \sum_{\ell=\ell_0}^{L} \sigma_\ell \sqrt{w_\ell}$$

Next we compare the MLMC work for a coarsest level $\ell_0$ with the MLMC work for a coarsest level $\ell_0' = \ell_0 + 1$. The level $\ell_0'$ is a better choice for the coarsest level than $\ell_0$ if

$$W_{\text{tot}}(\ell_0') < W_{\text{tot}}(\ell_0).$$

Using the formulas (6) and (7) for variance and work of the coarsest and the finer levels differ, this comparison simplifies to

$$\sigma_{\ell_0}^I \sqrt{w_{\ell_0}^I} + \sigma_{\ell_0+1}^I \sqrt{w_{\ell_0+1}^I} + \sigma_{\ell_0}^\Delta \sqrt{w_{\ell_0}^\Delta} > \sigma_{\ell_0+1}^I \sqrt{w_{\ell_0+1}^I}.$$
In order to decide if $\ell_0$ or $\ell'_0$ is the better lowest level it suffices to know the work $w_\ell$ and the variance $\sigma^2_\ell$ on these two levels. This allows us to propose a simple iterative algorithm to find an optimal $\ell_0$.

**Algorithm 6.1** Find an optimal coarsest level $\ell_0$

Given work $w_\ell$, variance $\sigma^2_\ell$, a fine level $L \geq \hat{L}$ to ensure termination of the algorithm and a very coarse level $\ell_0$. Find an optimal coarsest level $\ell_0$.

1: while $\ell \leq \hat{L}$ and $\sigma^I_{\ell_0} \sqrt{w^I_{\ell_0}} + \sigma^I_{\ell_0+1} \sqrt{w^I_{\ell_0+1}} > \sigma^I_{\ell_0+1} \sqrt{w^I_{\ell_0+1}}$ do
2: Increment $\ell_0$ by one.
3: end while

This algorithm ensures that $W_{\text{tot}}(\ell_0 - 1) > W_{\text{tot}}(\ell_0)$ and $W_{\text{tot}}(\ell_0) \leq W_{\text{tot}}(\ell_0 + 1)$ for $\ell_0 < \hat{L}$, which proves that the coarsest level $\ell_0$ is (locally) optimal. Note that a standard Monte Carlo method should be used instead of the MLMC method if $\ell_0 \geq L$.

7. FT-MLMC on a computer with unknown failure distribution

We propose a fault tolerant Multilevel Monte Carlo (FT-MLMC) method along the lines of Section 3.4, where the sample numbers and levels are optimized based on unknown parameters. The proposed FT-MLMC method will tolerate hard faults [16]. We disregard soft faults. Hard faults lead to an early termination of the computation and samples affected by a hard fault cannot be used in the final result. Such a lost sample may be recomputed or disregarded.

Failures are statistic in nature, resulting in a statistical work per successfully computed sample. We denote the expected work per successfully computed sample by $E[w_\ell]$. We use the inequality

$$E[w_\ell] \leq p_\ell w^{\text{fault free}}_\ell$$

with

$$p_\ell = E\left[\frac{M_\ell + M^{\text{failed}}_\ell}{M_\ell}\right],$$

where $M^{\text{failed}}_\ell$ are the number of unsuccessful attempts to compute a sample and $M_\ell$ the number of successfully computed samples. In the above equations we do account for the fact that the work for an unsuccessful attempt may be smaller than for a successful computation of a sample. We use $w^{\text{failed}}_\ell \leq w^{\text{fault free}}_\ell$ where $w^{\text{failed}}_\ell$ is the work of an unsuccessful attempt. Note that simple approaches as $E[w^{\text{failed}}_\ell] = 0.5 w^{\text{fault free}}_\ell$ only hold for time invariant failure distributions. This equality does not hold, e.g., for a Weibull failure model, a model that was suggested by Schroeder and Gibson for the time between failures of the publicly available failure logs [5].

The algorithm in Section 3.4 estimates the parameters on level $L + 1$. We propose to estimate the $p_{L+1}$ and hence the expected work $E[w_\ell]$ on level $L + 1$, by

$$p_{L+1} \approx \frac{w^{\text{fault free}}_{L+1}}{w^{\text{fault free}}_L}.$$  

Using this formula it is possible to estimate $w_{L+1}$ based on the measured (or a priori known) parameter $w^{\text{fault free}}_L$ in the “fault free” case.

With this procedure it is possible to use the algorithm sketched in Section 3.4 as an FT-MLMC method running on a computer with unknown failure distribution. It estimates optimal $M_\ell$ and $L$.
taking into account the experienced failures while the samples and the MLMC result are computed. A final FT-MLMC tolerance or a total work budget is respected by the algorithm. As the finest samples are most vulnerable to failures, the algorithm automatically reduces the number of levels, by dropping the few finest levels, in case of high failure rates.

8. Numerical experiments

8.1. Model problem

We demonstrate the ability of our approach using a model problem inspired by a time dependent two-dimensional \((d = 2)\) stochastic Euler equation of gas dynamics [17] where the samples are solved by a finite volume method with convergence rate \(s = 1\). The finite volume method uses a mesh width \(h_{\ell}\) on level \(\ell\) that determines the work needed to compute a sample as well as its discretization error. The mesh widths satisfy \(h_{\ell} = 1/2 h_{\ell-1}\) with \(h_0 = 1\). This leads to the following parameter for the work per sample

\[
\begin{align*}
  w^{\text{fault free}}_{\ell} &\sim h_{\ell}^{-(d+1)} = 2^{3\ell}, \\
  \tau_{L} &= h_{L}^s = 2^{-L}.
\end{align*}
\]

The level variance converges as well with \(h_{\ell}^s\), hence we set

\[
\sigma_{\ell} = \begin{cases} 
  \sigma_{\ell}^\Delta = h_{\ell}^s = 2^{-\ell}, & \text{if } \ell > \ell_0, \\
  \sigma_{\ell}^I = 0.1, & \text{if } \ell = \ell_0.
\end{cases}
\]

Note that choosing the coarsest level as \(\ell_0 = 0\) would result in a variance which is smaller on the coarsest level than on the next finer one \((\sigma_0 < \sigma_1^\Delta)\). Algorithm 6.1 will provide a better choice for \(\ell_0\), which results in \(\sigma_0 > \sigma_0^I\). As soon as (hopefully) optimal values for \(\ell_0\), \(L\) and \(M_{\ell}\) are found, the MLMC error is estimated using (5).

8.2. Implementation

We decided to first evaluate the capabilities of FT-MLMC, without a real fault tolerant program and with a simple failure model, to pave the way for a fault tolerant implementation e.g. using User Level Failure Mitigation (ULFM) [18] a fault tolerant version of MPI. In our experience (from [3]) such a fault tolerant ULFM implementation is challenging on its own, and it is beneficial if the algorithms in mind are previously tested. In the long run we aim at a fault tolerant client/server implementation, where the algorithm in Section 3.4 is executed on replicated clients and the samples are computed on multi-core servers. We evaluated different options of such a client/server implementation using ULFM in [19].

Now we discuss the fault intolerant implementation used to obtain the results presented in this paper. The absence of real fault tolerance demands a failure simulation. For simplicity, we use a fixed failure rate \(\lambda\) for a unit time interval, leading to the success probability of a sample of \((1 - \lambda)w^{\text{fault free}}_{\ell}\). A choice of \(\lambda = 10^{-8}\) would hence imply that an attempt to compute a sample on level 0 succeeds with a probability of \(1 - 10^{-8}\) and fails with a probability of \(10^{-8}\). On level 10, where the cost to compute a sample is much higher \(w^{\text{fault free}}_{10} = 2^{30} \approx 1.1 \cdot 10^9\), the computation of a sample succeeds with a probability of \(\approx 2.2 \cdot 10^{-5}\). This illustrates the fact that fine samples are much more vulnerable to failures than samples on the coarsest level.
This simple model has some advantages. Assuming perfect scalability the success probability remains independent of the number of processes involved in the computation of a sample. Additionally the success probability is independent of the starting time. More accurate failure models, as for instance the Weibull failure distribution fitted to experienced failures [5], lack these nice properties. As a consequence the parallelization as well as the sequence of the computation of samples would matter, such that both this additional parameters would have to be described and cleverly chosen. Additionally, the interpretation of the results might have been more complicated as additional parameters would have been involved. Therefore we use this simple failure model.

With the above failure model, the bound in equation (16) on the expected work per successfully computed sample becomes

$$\mathbb{E}[w_\ell] \leq \mathbb{E}[p_\ell] w_\ell^{\text{fault free}} = \mathbb{E} \left[ \frac{M_\ell + M_\ell^{\text{failed}}}{M_\ell} \right] w_\ell^{\text{fault free}} = \mathbb{E} \left[ \frac{1}{(1 - \lambda)^{w_\ell}} \right] w_\ell^{\text{fault free}}$$

(17)

For reference measurements we use this known bound on the expected work per successfully computed sample.

In the measurements, where the failure rate is assumed unknown, the bound in equation (17) cannot be used. Instead we base all our estimates on the measured numbers of successfully completed ($M_\ell$) and failed ($M_\ell^{\text{failed}}$) samples during an MLMC run. With the obtained sample numbers ($M_\ell$ and $M_\ell^{\text{failed}}$) we estimate

$$p_\ell \approx \frac{M_\ell + M_\ell^{\text{failed}}}{M_\ell}$$

as well as the expected work

$$\mathbb{E}[w_\ell] \approx p_\ell w_\ell^{\text{fault free}} = \frac{M_\ell + M_\ell^{\text{failed}}}{M_\ell} w_\ell^{\text{fault free}}.$$

In this paper, we will not run a real FT-MLMC method, nor compute a MLMC result. We will base our FT-MLMC evaluation only the on level variance $\sigma_\ell$, the discretization error $\tau_L$ (both defined in the previous section) and on the simulated expected work per successfully computed sample $\mathbb{E}[w_\ell]$. The resulting MLMC error is not obtained by performing an MLMC simulation, but is only estimated by evaluating equation (5). This has the clear advantage that no MLMC simulation has to be performed, which allows a faster evaluation without the use of high performance computing.

8.3. Results

In this section we execute four different experiments. The first one is entitled “discrete optimum” which represents the reference, where the failure distribution is known. Hence the bound (17) on the expected work per sample $\mathbb{E}[w_\ell]$ is known as well. As the name reveals optimal $L$ and $M_\ell$ are computed, such that the expected total work or the MLMC error is minimized, cf. Algorithm 3.1.

To quantify the influence of suboptimal choices of sample numbers $M_\ell$ and finest level $L$ we introduce the two measurements, “constant L” and “rounded continuous optimum”, where the exact failure distribution is known as well. Both compute the continuous optimal $M_\ell$ instead of the discrete optimum, and then round to integer values such that the corresponding constrain is respected. In the continuous optimization (Algorithms 4.1 and 5.1) the finest level $L$ is a parameter. In the “rounded continuous optimum” measurement we use the optimal $L$ from the “discrete optimum” algorithm. (In our measurements $L$ varies from $\ell_0 + 7$ to $\ell_0 + 5$). In the
“constant L” measurement the finest level remains constant independent of the failure probability. In our measurements $L = l_0 + 7$. This will indicate the influence of badly chosen $L$.

The last measurement “unknown computer” is the one we intend to implement in the future as our FT-MLMC method. We pretend that the failure distribution of the computer is not known, and base the estimates on the experienced failures. The “unknown computer” algorithm is described in Section 3.4. We use it with the parameters given in Section 7.

Discrete optimum
Constant L
Rounded continuous optimum
Unknown computer

Figure 2: Comparing different algorithms to minimize the expected work for a given MLMC tolerance.

Figure 2 compares these four algorithms for a fixed MLMC tolerance $\varepsilon$. All results are plotted against the failure parameter $\lambda$ which determines the failure distribution from the fault free case $\lambda = 0$ to an increased failure parameter of $\lambda = 2 \cdot 10^{-8}$ or $5 \cdot 10^{-8}$. As already mentioned the tolerance $\varepsilon$ is fixed, hence when solving the optimization problems the expected work $\mathbb{E}[W_{tot}]$ should be minimized. Figures 2(a) and 2(c) show that the expected work increases with the failure parameter $\lambda$ in order to maintain a certain tolerance $\varepsilon$. The expected work is normalized to
the fault free execution of the “integer optimum” algorithm. We observe that in Figure 2(a) all algorithms perform similarly except “constant L” for which the performance drops significantly for larger failure rates. This is expected, as the finest level $L = \ell_0 + 7$ remains constant independent of the failure rate. For large failure rates it becomes almost impossible and hence very expensive to successfully compute a sample on level $L$. Figure 2(b) shows how the other approaches adapt their finest levels to the increasing failure rate. For the “discrete optimum” algorithm a certain failure parameter $\lambda \approx 0.4 \cdot 10^{-8}$ exists from which on the optimization only uses $L = \ell_0 + 6$ levels.

The “unknown computer” algorithm finds optimal values for $M_L$ and $L$ based on experienced failures. As random failures appear it is clear that the resulting $M_L$ and $L$ are as well random to a certain extent. This means that e.g. for $\lambda = 0.4 \cdot 10^{-8}$ the “unknown computer” algorithm sometimes uses $L = \ell_0 + 6$ and sometimes $L = \ell_0 + 7$. In Figure 2(b) we present therefore the expected finest level $E[L]$ for the “unknown computer” algorithm, approximated over 50 repetitions of the “unknown computer” algorithm for each measured $\lambda$. The curve is still slightly fluctuating, which could by cured by even more repetitions. It is observed that for small $\lambda \lesssim 0.1 \cdot 10^{-8}$ the algorithm always chooses $L = \ell_0 + 7$ and for large $\lambda \gtrsim 0.6 \cdot 10^{-8}$ always $L = \ell_0 + 6$. In between both choices are found, as expected. The effect of mischoosing $L$ has effects on the expected work in Figure 2(a). Mainly visible at $0.4 \cdot 10^{-8} \lesssim \lambda \lesssim 0.6 \cdot 10^{-8}$ where the “unknown computer” algorithm shows increased and more oscillatory results. At around $\lambda = 0$, where no failures occur, the “unknown computer” algorithm performs more poorly than all the others. This is due to the relative large number of start-up samples ($K = 20$) used for this plot. This forces the algorithm to compute too many samples on some levels. In contrast the same measurements where performed with $K = 10$ in Figure 2(c) and 2(d), respectively. Figure 2(c) shows the down side of using a small $K$, as the decision of adding a new level is made with inadequate data. Therefore the “unknown computer” algorithm under-performs for $0.4 \lesssim \lambda \lesssim 0.9 \cdot 10^{-8}$. We observe that the choice of the parameter $K$ in the algorithm of Section 3.4 is rather delicate. It may be improved by choosing a rather small $K$ as long as few failures are observed, and use larger $K$ if many failures war observed.

We fix a work budget and compare the results of the four algorithms, which try to minimize the expected MLMC error. In Figure 3 we observe that the “constant L” algorithm performs worse than all other algorithms. This results from a fixed finest level which is not adapted to the failure rate. Around $\lambda \approx 2.2 \cdot 10^{-8}$ the “discrete optimum” algorithm initially shows a strong local error increase. Afterwards the expected error remains more or less constant as soon as $L$ is decremented by 1. The “rounded continuous optimum” algorithm finds optimal real $M_L$ and then rounds them down to the next integer, to ensure that the effective work stays below the work budget. The rounding significantly affects the result at $\lambda \approx 2.1 \cdot 10^{-8}$ where $M_L$ is rounded down to 0, such that the finest level is decreased by 1. This increases the MLMC error, with respect to the “discrete optimum” algorithm by approximately 30%.

In the “unknown computer” algorithm three start-up samples ($K = 3$) were used. In Figure 3 we observe that even for small $K$ it performs well. Between $1 \cdot 10^{-8} \lesssim \lambda \lesssim 2.2 \cdot 10^{-8}$ the performance of the “unknown computer” algorithm cannot keep up with the “discrete optimum”, the penalty being around 25%, which appears to be acceptable. In general we see convincing results, where the expected error only increases moderately by a factor of 2 starting from no failures.

Figure 3(b) shows how the finest level $L$ adapts to failures. Both algorithms, “unknown computer” and “discrete optimum”, start with $L = \ell_0 + 7$ for $\lambda = 0$ and reduce $L = \ell_0 + 5$ for larger $\lambda \approx 5 \cdot 10^{-8}$. In between the accordance is not very good. However, as discussed the MLMC error is only increased by around 25%. In Figure 3(b) measurements are shown up to $\lambda = 5 \cdot 10^{-8}$. For
larger failure parameters $\lambda$ the finest level would further drop to $L = \ell_0$ which corresponds to plain Monte Carlo on the coarsest mesh.

9. Conclusion

We presented algorithms to determine optimal MLMC parameters. Using a branch-and-bound algorithm we determined integer optimal sample numbers $\{M_\ell\}$ as well as the optimal finest level $L$. Also a simple algorithm was introduced to determine an optimal coarsest level $\ell_0$.

The proposed branch-and-bound algorithm terminates within a few seconds. This is negligible when compared with common MLMC simulations that require minutes up to hours on parallel computers. Using optimal values for $M_\ell$, $\ell_0$ and $L$ can significantly reduce the MLMC execution time.

We propose a fault tolerant MLMC method where samples are recomputed if they were lost due to failures. With a high rate of hard failures it is nearly impossible to successfully compute very fine samples, and hence the expected cost to successfully compute them is immense. Solving an integer optimization problem which takes the failure rate into account avoids this problem as it will lower the finest level $L$ if needed. We have measured that the adaptation of the MLMC parameters to the failure rate improves the MLMC error versus work ratio dramatically. Additionally we presented a method in which the failure rate is estimated based on failures experienced during the computation. This allows our fault tolerant MLMC method to be applied even in cases where the failure rate of the computer is not known in advance. We are optimistic, that the promising results obtained in this paper can be confirmed with a real fault tolerant MLMC implementation.
References


