Solution Bundles of Markov Performability Models through Adaptive Cross Approximation

Giulio Masetti*, Leonardo Robol†*, Silvano Chiaradonna*, Felicita Di Giandomenico*
* ISTI-CNR, Pisa, Italy,
† University of Pisa, Italy,
{giulio.masetti, silvano.chiaradonna, felicita.digiandomenico}@isti.cnr.it,
leonardo.robol@unipi.it

***Abstract*—A technique to approximate solution bundles, i.e., solutions of a parametric model where parameters are treated as independent variables instead of constants, is presented for Markov models. Analyses based on an approximated solution bundle are more efficient than those that solve the model for all combinations of parameters' values separately. In this paper the idea is to properly adapt low rank tensor approximation techniques, and in particular Adaptive Cross Approximation, to the evaluation of performability attributes. Application on exemplary case studies confirms the advantages of the new solution technique with respect to solving the model for all time and parameters' combinations.**

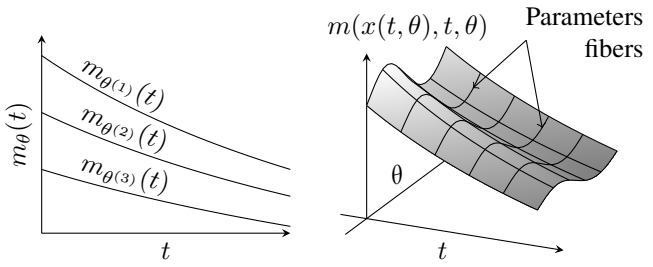
A. Acronyms and Symbols

ACA	Adaptive Cross Approximation
CTMC	Continuous Time Markov Chain
GLM	Generalized Linear Model
LU	Lower Upper Factorization
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
UPS	Uninterruptible Power Supply
F	A set of ordinary differential equations
k	Number of approximants
m	Measure
M	Tensor of the measure
$\hat{m}$	Approximated measure
$\hat{M}$	Approximated tensor of the measure
N	Number of the model states
p	Number of parameters
θ	Parameters vector
x	Solution of $F = 0$
π	State probability vector
b	Expected sojourn time vector
Q	Infinitesimal generator matrix
r	Reward vector
$\mathcal{S}$	State-space
V	Instantaneous performability measure
Y	Accumulated performability measure
n_{sample}	Number of samples
n_{batch}	Number of simulation batches
n_j	Number of approximation points on the j-th fiber
λ	Both case studies: failure rate
μ	Both case studies: component repair rate
Reliability	Both case studies: reliability
$t_{\max}$	Both case studies: mission time
c	Case study 1: component failure coverage
c_f	Case study 1: component failure coverage
c_r	Case study 1: system repair coverage
μ_d	Case study 1: system repair rate from standby
n_r	Case study 1: number of components
Under_repair	Case study 1: mean percentage of time where at least one component is operational and at least one component is under repair
B	Case study 2: charge level of the battery
L	Case study 2: rundown time of the battery
Low_charge	Case study 2: mean (or expected) percentage of time where the battery is low
n_c	Case study 2: maximum number of battery destructive discharging cycles

I. Introduction

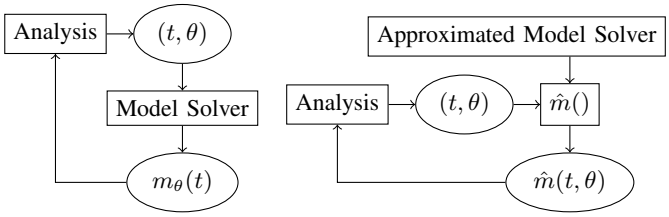
Stochastic model-based analysis is widely adopted to study dependability, performance and performability attributes of complex systems, in particular computing and communication ones. At a certain level of abstraction, such systems can be modeled by Continuous Time Markov Chain (CTMC) [1], where a reward is attached to each state to support the measures of interest in terms of reward measures [2], [3].

Here, the focus is on solving the CTMC to evaluate instant-of-time (at a finite time) and interval-of-time (relative to an interval of finite length) measures [1], [3] for systems of increasing largeness and complexity. Addressing these systems is a challenge for classical solution methods and a test bench for new ones. In addition, accuracy of results from model-based analysis strongly depend on the model parameters setting, and it is often required to solve the model for different time and parameters assignments (from now on called *points* in the Cartesian product of time and parameters' space) so further exacerbating the difficulties encountered in solving the model. Closed formulas are infeasible to be found for large models, and both classical alternatives to model solution, i.e., numerical analysis- and simulation- based ones, face limitations when employed as solution subroutines. The main issue is originated by passing points one at a time to the solution subroutine, as shown in Figure 2a, where the model solver is called on each point of (t, θ) to get the measure $m_\theta(t)$.



(a) Parameters as constants (b) Solution bundle

Fig. 1. Different treatment of parameters: (a) as constants, (b) as variables. The proposed application of ACA exploits efficient computational strategies to evaluate measures along time and parameters' fibers.



(a) On demand approach (b) Decoupled approach

Fig. 2. (a) standard approach, where the solver is called on each sample of constant parameters (t,θ) to get the measure $m_\theta(t)$; (b) decoupled approach, where first the measure approximation $\hat{m}$ is determined and then exploited.

A step forward is to consider the semi-symbolic method presented[1] in [7], where time is symbolic and all the rest is a numeric value, as depicted in Figure 1a, where the measure $m_\theta(t)$ is evaluated for each specific value $\theta^{(1)}, \theta^{(2)}$ and $\theta^{(3)}$ of the parameter θ. This way, continuity in time is maintained in the subroutine call, but alone it is insufficient to solve the issue because combinations of parameters' values are still passed one by one.

The natural generalization is to transform the role of parameters from "constants yet to be defined" to continuous variables (details in Section II-B) and then defining the measures of interest as continuous functions of both time and parameters. Borrowing terminology from physics, solving a model where parameters are made variables produces a *solution bundle* [8], as depicted in Figure 1b, i.e., solutions $x(t,\theta)$ of a parametric model where the parameters t and θ are treated as independent variables instead of constants. Therefore, the focus of this paper is on *measure bundles*, represented in Figure 1b by all the values that the measure $m(x(t,\theta),t,\theta)$ can assume for every possible value of t,θ and $x(t,\theta)$.

Clearly, expressing exactly such functions is equivalent to write closed formulas, and then it is infeasible in practice, but this shift of point of view allows to *approximate* the measures, and then to radically simplify data flow. Now, as depicted in Figure 2b, analysis and model solution logics are decoupled: first an approximation $\hat{m}()$ of the measures is determined, and then the approximation can be consumed by analyzers, that are also able to exploit the smoothness of the approximation (see Section II-B), to get $\hat{m}(t,\theta)$.

Several techniques are available to perform approximation. Roughly simplifying a complex topic, all the approximation methods require to evaluate the measures accurately on a (small) set of points and to tweak a pre-defined set of approximation functions (called approximants) that, combined together, define continuous approximations of the measures.

In this paper, the Adaptive Cross Approximation (ACA) technique is adapted to approximate performability measure bundles, that in turn are defined on solution bundles of the Kolmogorov forward equation that characterizes CTMCs. In particular, ACA is based on *separable approximants*, i.e., each approximant is the product of functions that depend only on time or on a single parameter. Arbitrary accuracy of the technique has been proven in [9].

Therefore, the main contribution of this paper is the development of an efficient and accurate method to evaluate performability measures along fibers, exploiting the separable approximants feature of ACA, as depicted in Figure 1. It works well under the assumption that the dependencies of the model on parameters are smooth functions (i.e., have continuous derivatives up to a reasonably high order), as exemplified in Section II-C. Notice that time is one of the fibers, thus the proposed technique can be interpreted as a generalization of the semi-symbolic solution method presented in [7]. Details are in Sections IV-D and IV-E.

Enlarging the view and considering that Approximation Theory and Machine Learning are close disciplines, it is possible to classify results of the former adopting the perspective and the parlance of the latter. In particular, the presented approach can be classified as *Interpretable AI* [10] because separable approximants promote *decomposability*, a key aspect of IAI.

The rest of the paper is organized as follows. Section II provides the background knowledge on Markov Dependability (mainly Reliability-focused) models, reward structures and measures of interest. To better appreciate the novelty of the contribution, a high level description of the problem is also provided. Section III briefly discusses how to choose approximation points. Section IV presents the new method. Section V introduces the two case studies, then exploited in Section VI to conduct performance analysis and comparison with two alternative methods. Section VII reviews related work. Section VIII draws conclusions and discusses future work.

II. Context and Contribution

In this section the addressed parametric models and measures of interest are introduced. Then the problem of the evaluation of such measures for different values of the parameters is described in abstract terms to better focus on the shift in role of the parameters and highlight the issues that have to be solved or mitigated. Finally, the kind of parameters that can appear in the models are discussed.

In the following, subscript θ indicates that parameters are treated as constants, while θ in parenthesis indicates that they

[1]Originally thought to address a different problem, namely largeness vs stiffness vs accuracy [4]–[6], but applicable here to mitigate the issue.

are treated as variables. Moreover, all the vectors are row vectors, following the Markov chain community notation.

A. Models and measures of interest

The models addressed in this paper are CTMCs with state-space $\mathcal{S}$ and with infinitesimal generator matrix $Q(\theta)$, where $\theta \in \mathbb{R}^p$ is the parameters vector of the model. This means that some of the entries of Q are functions of some of the entries of θ. At every instant of time t, also the state probability vector $\pi(t,\theta)$ depends on θ. The equations that characterize the CTMC and the measures of interest are:

$$\begin{cases} \frac{\partial \pi(t,\theta)}{\partial t} & = \pi(t,\theta) \cdot Q(\theta), \\ \pi(0,\theta) & := \pi_0, \end{cases} \tag{1}$$

$$\begin{cases} \frac{\partial b(t,\theta)}{\partial t} & = b(t,\theta) \cdot Q(\theta) + \pi_0, \\ b(0,\theta) & := 0, \end{cases} \tag{2}$$

where π_0 is the initial probability vector, independent of θ, and $b(t,\theta) = \int_0^t \pi(\tau,\theta)d\tau$, i.e., the i-th entry of $b(t,\theta)$ is the sojourn time of the CTMC in the state i in the interval $[0,t]$.

Unfortunately, closed formulas for the solution of Equations (1) and (2) are not directly exploitable for concrete computations [1] when the models at hand are too large to allow explicit matrix powers evaluation and, even for relatively small models, when reliability/availability models are *stiff*[2]. This motivates the quest for solution methods capable to address large CTMC and to tolerate stiffness.

Measures of interest addressable by the proposed solution bundle method are in general dependability, performance and performability indicators. For the sake of simplicity, in the following performability is referred as the representative measure, with other indicators introduced only when specifically addressed.

Performability measures are defined in terms of moments of the following random variables:

$$V := \sum_{s \in \mathcal{S}} r_s \cdot I_t^s, \tag{3}$$

$$Y := \sum_{s \in \mathcal{S}} r_s \cdot J_{[0,t]}^s, \tag{4}$$

where I_t^s is the indicator random variable representing the event that the model is within state s at time t, $J_{[0,t]}^s$ is the random variable representing the total time the model is within state s during $[0,t]$. Here, r_s is a real value gained by the stochastic process while staying within the state s. Correspondingly, r will be called *reward vector*.

Starting from the definition of expected value and switching the order of integration, Equation (1) and Equation (3) can lead to define the instant of time reward measure $\mathbb{E}[V] = \mathrm{dot}(\pi, r)$. Similarly Equation (2) and Equation (4) lead to the accumulated reward measure $\mathbb{E}[Y] = \mathrm{dot}(b, r)$. Other performability measures can be defined similarly in terms of higher moments or as conditional expectations [1], [3].

[2]Extreme disparity among the entries of the infinitesimal generator matrix and the time horizon of interest [11]. More formally, calling $q(\theta) := \max_{ij} |Q_{ij}(\theta)|$, if $q(\theta) \cdot t_{\max}$ is large then the model is stiff.

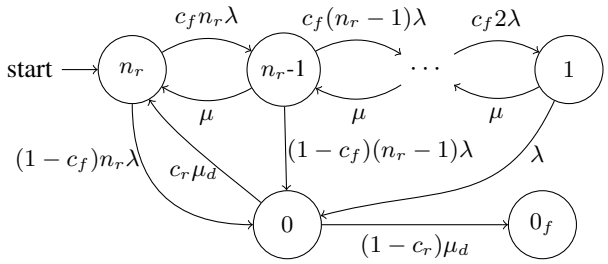


Fig. 3. Running example: CTMC of Case study 1, representing a degradable system with failure coverage and repair. Each state $i \geq 0$ represents the number i of currently operational components; 0_f represents the system failure. Full description is in Section V-A.

If the reward vector has nonnegative entries, then the measures can be evaluated numerically maintaining the accuracy obtained for the probability vector. In fact, V and Y are sums of positive values, thus avoiding any numerical cancellation. Otherwise, it is important to avoid the computation of differences of large numbers, which can be achieved by re-scaling the rewards.

Finally, to exemplify the contribution of Section IV when appropriate, consider the running example in Figure 3, where the CTMC of Case study 1 is illustred (full description is in Section V-A). For exemplification purpose, only one parameter $\theta_1 := \lambda$ is considered, while all the other parameters are defined as constant numerical values.

B. Problem Characterization and Contribution

Within a Reliability or Availability model [1], p parameters, i.e., the entries of the vector $\theta \in \mathbb{R}^p$, are p constants yet to be defined. So, in general, the measure of interest $m_\theta(t)$ is a function of $x_\theta(t) \in \mathbb{R}^N$, that in turn solves the equation $F_\theta(x_\theta(t)) = 0$, where N is the number of model states.

In the context of this paper, the abstract function m and the variable x correspond to the concrete reward measures $\mathbb{E}[V]$ or $\mathbb{E}[Y]$ and to π or b, respectively. In the running example, m is the Reliability and θ is the scalar λ shown Figure 3. Moreover, F can be the forward Kolmogorov equation or the integral Kolmogorov equation corresponding respectively to Equations (1) and (2).

In both cases, F_θ is a set of Ordinary Differential Equations (ODEs). This way, a family of equations and solutions, indexed by the constant θ, is defined, each having t as independent variable and the components of $x_\theta(t)$ as dependent variables.

To evaluate how θ affects $m_\theta(t)$, classical methods first define the samples $\theta^{(1)}, \ldots, \theta^{(n_{\mathrm{sample}})}$, drawn from some distribution, and then solve n_{sample} times $F_\theta = 0$ or produce n_{batch} results for each sample (total of $n_{\mathrm{sample}} \cdot n_{\mathrm{batch}}$ simulations) directly simulating[3] the model (as shown in Figure 2a). From one hand, statistical analysis grounded on model simulations is perfect for assessing the impact of parameters' uncertainty, but

[3]A word of caution: in the ODEs community, *simulation* usually refers to numerical solution of the equations. Instead in this paper "simulation" means "discrete event simulation", as usually understood in the context of discrete space, continuous time models.

can rarely achieve high accuracy when dealing with large and complex models, and sometimes addressing rare events can be problematic. On the other hand, numerical analysis typically offers an accurate solution for each sample of θ, and insights on parameters' relevance through local sensitivity analysis, but the other kind of analysis (e.g., propagating uncertainty from parameters to results) is sometimes infeasible because the solution of each instance can require huge computations, and a large n_{batch} is needed.

Conversely, in this paper the parameters θ are treated as independent variables instead of constants, so the set of Partial Differential Equations $F(x(t,\theta),\theta) = 0$ defines the *solution bundle* $x(t,\theta)$, that in turn defines the *measure bundle* $m(x(t,\theta),t,\theta)$. The shift of parameters' role is depicted in Figure 1. In the following, to simplify notation, the arguments of m will be omitted when clear from the context.

Once an expression for m is found, local sensitivity analysis becomes (following one of the most popular definitions [12]): m is sensitive to a (small) variation of θ_j if $\partial m/\partial\theta_j$ is large. Similarly, global sensitivity analysis, uncertainty quantification [13], i.e., study how uncertainty in the evaluated measure can be attributed to different sources of uncertainty in parameters value, parameter estimation [14] and optimization [15] can all be expressed in terms of m. Also, computations data flow is simplified, as depicted in Figure 2. The problem is that finding a closed formula for m is infeasible in practice for large N.

Thus, the presented method exploits classical Approximation Theory, where a set of functions (dense in the solution space) are employed to obtain an approximation $\hat{m}(t,\theta)$ of $m(x(t,\theta),t,\theta)$. Notice that $\hat{m}$ does not depend directly on x, as m does. Depending on which aspect is intended to be highlighted, and the community where a particular technique is imported to, such an approximation is called differently. For instance, in Machine Learning, being the focus on input-output relations, m itself is called *model*, and, being the approximation defined starting from a few evaluations of the model (details in Section IV), $\hat{m}$ is called *metamodel* [16], [17]. When highlighting the fact that, to be useful, the approximation has to be much simpler to evaluate than the original at the cost of loosing in accuracy, $\hat{m}$ is called *surrogate model*. Here, to better distinguish the Reliability/Availability model from its solution, $\hat{m}$ will be called *approximated measure bundle*. The main issues encountered when addressing *how* to define $\hat{m}$ are:

- the fully symbolic evaluation of m, in t and θ, is unfeasible for large models (pictorially: moving freely on all the gray area in Figure 1b),
- the semi-symbolic method treats symbolically only time (pictorially: moving only along time fibers in Figure 1b), and the already published generalizations are:
  - designed to work only for $p = 1$, such as `chebop2` (that is unacceptably restrictive because it works only for $N = 1$ [18]),
  - too computationally expensive[4],
  - feasible for acyclic CTMC (e.g., generalization of [19]),
- fully-numeric methods are computationally too expensive, as discussed at the beginning of this section, because constructing first the full tensor (i.e., the multidimensional array) with entries

$$M_{i_0,i_1,\ldots,i_p} := m\big(t^{(i_0)}, \theta_1^{(i_1)}, \ldots, \theta_p^{(i_p)}\big) \tag{5}$$

for samples indexed by $i_0 \in \{1,\ldots,n_0\}$, $i_1 \in \{1,\ldots,n_1\}$,..., $i_p \in \{1,\ldots,n_p\}$ and then defining $\hat{m}_{t,\theta}$ *through interpolation* on M is unfeasible: it requires $\mathcal{O}(n_0 \cdot n_1 \cdots n_p)$ solutions of $F = 0$ and huge memory.

Thus, here, by the adaption of the ACA algorithm first proposed by Bebendorf in [9], $\hat{m}$ is an approximation of m defined as a sum of *separable* approximations

$$\begin{aligned}\hat{m}(t,\theta) :=& f_1(t)\cdot g_{1,1}(\theta_1)\cdots g_{1,p}(\theta_p) + \ldots \\ &\cdots + f_k(t)\cdot g_{k,1}(\theta_1)\cdots g_{k,p}(\theta_p),\end{aligned} \tag{6}$$

where f_i and g_{ij} are approximation functions chosen as detailed in Section III. The approximation is separable in the sense that, for all i and j, g_{ij} depends only on one parameter (and f_i only on time), so pictorially corresponds to moving along all the fibers in Figure 1b. This makes the method interpretable. In addition, the great improvement with respect to the full evaluation of the tensor as in Equation (5) is that, being the solution of the set of Partial Differential Equations (PDEs) $F = 0$ smooth, evaluating $\hat{m}$ on sample points requires $\mathcal{O}(k)$ operations each, and generates $\hat{M}$ of rank k, that can be stored efficiently (details in Section IV). Stated differently, finding $\hat{M}$ means to find a low-rank approximation of M. For $p = 1$, $\hat{M}$ is a matrix, and the theory behind the approximation is fully understood. For $p > 1$, theory is not able to answer key questions yet, as discussed in Section IV-C.

C. Dependencies on parameters

Several kinds of parameter can appear in performability models. Here, examples that appear often, and the dependency they introduce in $Q(\theta)$, are classified according to the corresponding elementary functions. Stated differently: each entry of $Q(\theta)$ can be a constant or a function of θ, the types of function that appear often in the addressed models are:

linear: e.g., $Q_{n_r,n_r-1}(\theta) = c_f \cdot n_r \cdot \theta_1$ in the running example (as shown in Figure 3),

quadratic: e.g., $Q_{ij}(\theta) = \theta_1 \cdot \theta_2$ and $Q_{ij}(\theta) = \theta_1 \cdot (1-\theta_2)$ when modeling failure coverage, i.e., $\theta_1 = \lambda$ is the failure rate and $\theta_2 = c$ is the coverage probability,

reciprocal: e.g., $Q_{ij}(\theta) = n_e/\theta_1$ when approximating a non-exponential sojourn time with an Erlang (in this case a deterministic time $\theta_1 = L$ approximated with an Erlang comprising n_e exponential jumps whose rates are all equal to n_e/L) and n_e is a constant.

[4]To the best of authors' knowledge, the method implemented in SHARPE [7] addresses only m_θ with complexity $\mathcal{O}(N^3)$. Extending it to address m could be too expensive for practical usability.

In all the mentioned cases, $Q(\theta)$ is smooth, and then the measure approximation $\hat{m}$ defined in Equation (6) is cheap to evaluate and store, as detailed in Section IV, where ad hoc methods to evaluate the fibers of $\hat{M}$ are presented. Notice that the method introduced in this paper can address, with slight modifications, other elementary functions as well (as long as m is smooth).

III. Choice of the evaluation points

The method proposed evaluates the measures of interest at a discrete set of times $t^{(i)}$, and at prescribed values for the parameters θ_j, for $j = 1, \ldots, p$.

For what concerns the time variable, the choice of evaluating at a uniform grid on the interval $[0, t_{\max}]$, where $t_{\max}$ is the system mission time, is often encountered, and may depend on the specific method used to integrate the equation.

If the values that are of interest for the parameters in the vector θ are known a priori, then it is very reasonable to choose these values for the discretization. Namely, if the points needed for the analysis are known in advance then it is reasonable to choose the approximation points as a (small) subset of those. Otherwise, if the parameter θ_j takes values in the interval $[\mathrm{lb}, \mathrm{ub}]$ a natural choice may be to consider discretizing

$$\theta_j^{(i_j)} := \frac{\mathrm{ub}+\mathrm{lb}}{2} + \frac{\mathrm{ub}-\mathrm{lb}}{2} \cos\left(\frac{\pi(i_j-1)}{n_j-1}\right),\ i_j = 1, \ldots, n_j, \tag{7}$$

which are often called *Chebyshev points* of $[\mathrm{lb}, \mathrm{ub}]$. Indeed, the representation in the Chebyshev basis of the polynomial interpolant on these points can be found efficiently using a discrete cosine transform in $\mathcal{O}(n_j \log n_j)$.

Classical results on the Lebesgue constant for Chebsyhev interpolation of a continuous function $[\mathrm{lb}, \mathrm{ub}] \to \mathbb{R}$ guarantee that this result is almost accurate as the best polynomial approximation on the interval, up to a factor that grows as $\log(n_j)$. In addition, if the function has higher regularity (as it is in this case assuming that Q depends smoothly on $\theta_1, \ldots, \theta_p$, as in Section II-C) a very quick convergence rate can be guaranteed. For further details, the reader is referred to [20].

In practice, the choice of points will have little to no influence on the proposed solution method (and this will become clear in Section IV-D). From now on, it is assumed that this choice has been made once and for all.

IV. Proposed Method

This section describes the numerical method used to construct a compressed representation of all the evaluations of the measure on a $(p+1)$-dimensional lattice.

More precisely, from now on it is assumed that the discretizations of the time variable t and the parameters $\theta_1, \ldots, \theta_p$ are chosen a priori as

$$t^{(1)} < t^{(2)} < \ldots < t^{(n_0)}, \quad \theta_i^{(1)} < \theta_i^{(2)} < \ldots < \theta_i^{(n_i)},$$

for $i = 1, \ldots, p$. The goal is to determine a compact (or compressed) representation of the $(p+1)$-dimensional tensor $\hat{M}$ in Equation (5) obtained evaluating the measure $m(t, \theta)$ at any time and parameter in the above set. The case where $m(t, \theta)$ is the instantaneous measure $\mathbb{E}[V(t, \theta)]$ based on Equation (3) is described first. The case where $m(t, \theta)$ is an accumulated measure based on Equation (4) is then obtained with minimal modifications, replacing the function that computes $\pi(t, \theta)$ with the one that computes $b(t, \theta)$. In presence of several parameters, it may be challenging to store all the entries of the tensor $\hat{M}$ if $n_0, \ldots, n_p$ are even modest values, as the cardinality of its entries grows exponentially with p.

The proposed compressed representation will only require $\mathcal{O}(n_0 + \ldots + n_p)$ storage (and complexity), where the hidden constant depends on the regularity properties of the multivariate function $m(t, \theta)$.

The key ingredient to proceed is being able to evaluate the so-called *solution fibers*, i.e., to evaluate the measure for all values of a single parameter, while the others remain fixed. The following two sections describe how to perform this for the time variable and for $\theta_1, \ldots, \theta_p$.

A. The solution along the time fiber

A broad family of methods is available for solving the Kolmogorov equations for fixed values of the parameters $\theta_1, \ldots, \theta_p$. Which one to use is not particularly relevant for the purpose of this work.

The method under consideration relies on being able to evaluate the so-called *fibers* of the tensor, where all but one index are fixed, much more efficiently than all the entries in the tensor. The case of the fiber with respect to the index i_0 is the simplest to describe, as this corresponds to computing the measure $m(t, \theta)$ at all times $t^{(i_0)}$ of interest, and is easily obtained leveraging one of the methods cited in Section VII.

B. Evaluating the parameter fibers

The other relevant case is to fix all but one parameter $\theta^{(i_j)}$, and it is not restrictive to assume that $j = 1$; hence, the objective is determining the value of $m(t^{(i_0)}, \theta_1^{(i_1)}, \ldots, \theta_p^{(i_p)})$ for all possible values of i_1. Using the explicit expression of the solution to Equation (1) one obtains the following expression:

$$\pi_0^T e^{t^{(i_0)} Q(\theta_1^{(i_1)}, \ldots, \theta_p^{(i_p)})} r.$$

Hence, evaluating the measure can be recast into computing the action of the matrix exponential e^{tQ} on a vector for multiple values of Q (the dependency of Q on the parameters is not made explicit in the next equations to simplify the notation). An accurate (and easily computable) approximation of this value can be obtained by replacing the matrix exponential with a rational function

$$e^{tQ} r \approx \xi(tQ) r = \sum_{h=1}^{k} \alpha_h (z_h I + tQ)^{-1} r, \quad z_h > 0.$$

and therefore restating the problem as solving k linear systems with shifted copies of Q. Since Q is an M-matrix with eigenvalues with negative real part, it is natural to choose an approximant $\xi(z)$ that guarantees $|\xi(z) - e^z| \leq \epsilon$ for any $z \in \mathbb{R}_-$, the negative real semiaxis, for a properly chosen ϵ. If the eigenvalues of Q are sufficiently close to the real axis, a

low error ϵ can be rigorously guaranteed through the use of Bernstein ellipses (see [20] for the details, whose discussion is beyond the scope of the manuscript).

An appropriate choice for the approximant ξ, i.e., a choice of $\alpha_h \in \mathbb{R}$ and $z_h \in \mathbb{C}$ for $h = 1, \dots, k$, is given by the rational function obtained using Gonchar poles [21], which guarantee a convergence with rate $\epsilon \lesssim 9.28^{-k}$ (the development of such estimates has been long and difficult, and is well summarized in the review paper about the $1/9$-conjecture [21]). Precomputed versions of such poles through the Remez algorithm are used in this work for $1 \leq k \leq 16$, which allow to reach the same accuracy of double precision floating point arithmetic.

Obtaining the evaluations $m(t^{(i_0)}, \theta_1^{(i_1)}, \dots, \theta_p^{(i_p)})$ requires the solution of $n_1 \cdot k$ linear systems, which has the sought asymptotic complexity, but may have a non-negligible cost in practice. In Section IV-D we discuss how to further optimize this step.

C. Low-rank structure and adaptive cross approximation

It is now essential to note that the entries $M_{i_0,\dots,i_p}$ of the M tensor, defined in Equation (5), are the evaluations of a smooth (analytic) function over a $(p+1)$-dimensional lattice. This fact can be used to prove that M is well-approximated by a low-rank tensor, i.e.,

$$M \approx \hat{M} := \sum_{h=1}^{k} \frac{\hat{M}_{h,0} \otimes \dots \otimes \hat{M}_{h,p}}{\text{pivot}_h}, \tag{8}$$

where $\otimes$ denotes the tensor product [18], [22] and $\hat{M}_{h,j} \in \mathbb{R}^{n_j}$ corresponds to the evaluation of the j-th fiber of M.

It is useful to restrict the attention to the special case $p = 1$ first, as in the running example shown in Figure 3, where Equation (8) is just a low-rank factorization of the matrix M. In this case, several techniques are available to obtain an accurate factorization as in Equation (8) effectively (to name a few, roughly sorted from the more expensive to the cheapest: the SVD [23], QR factorizations with pivoting [24], randomized sketching [25], Golub-Kahan-Lanczos bidiagonalization [23]).

In general, it is convenient to consider a particular technique, named ACA, that allows to obtain a factorization as in Equation (8) by evaluating $\mathcal{O}(k)$ rows and columns of the matrix [26]. The procedure can be interpreted as a partial Lower Upper Factorization (LU) [23], and its convergence properties derived through this factorization.

In a nutshell, the ACA works as follows, for a sequence of steps $h = 1, 2, \dots, k$:

h.1) A sufficiently large (in modulus) entry in M is chosen, and is fixed as the current pivot.

h.2) The row $\hat{M}_{h,r}$ and the column $\hat{M}_{h,c}$ containing the pivot are used to construct the unique rank-1 approximation $\hat{M}_h := \hat{M}_{h,c} \cdot \text{pivot}^{-1} \cdot \hat{M}_{h,r}$ of M that coincides with the computed fibers on the selected row and column[5].

h.3) M is (implicitly) replaced with $M - \hat{M}_h$, and the procedure is continue as long as M is large enough.

[5]Note that, up to reshaping the rank-1 term, this is exactly $\hat{M}_{h,0} \otimes \hat{M}_{h,1}$ described in (8). $\hat{M}_h$ actually corresponds to $f_h(t) \cdot g_{h,1}(\theta_1)$ of Equation (6).

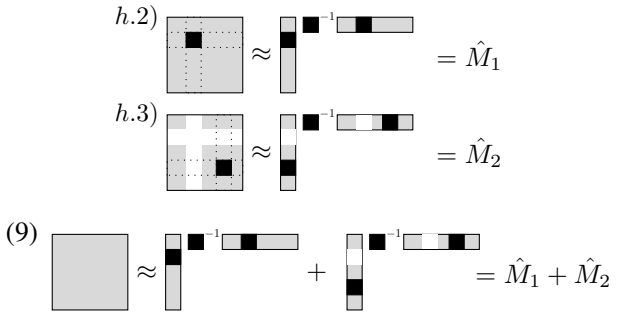


Fig. 4. The above picture represents two steps of ACA for $p = 1$. In $h.2$), the first pivot (the black square) is selected, and used in the rank-1 approximation in the right hand side. In $h.3$), the residual $M - \hat{M}_1$ is considered, which has a zero column and row; another pivot is chosen, and the two rank-1 contributions are combined in (9).

At the end, one has accumulated k rank-1 approximations $\hat{M}_1, \dots, \hat{M}_k$ such that

$$M \approx \hat{M}_1 + \dots + \hat{M}_k \tag{9}$$

which can be expanded in tensor products as in (8). The procedure is summarized pictorially in Figure 4. For the running example shown in Figure 3, the columns of M comprise measure evaluations where λ is fixed and time varies, and the rows those where time is fixed and λ varies.

The difficult part is choosing the pivot at item h.1) in a way that produces a stable algorithm. For instance, choosing the largest modulus entry in the current fiber as pivot makes the algorithm equivalent to LU with partial pivoting [18], and hence very stable in practice.

Generalizing this idea to more than two variables is difficult in general, since the theory is much weaker. In particular, the connection with the LU factorization with partial pivoting cannot be claimed any more, and the low-rank approximation problem for tensors is much more challenging than the one for matrices (and this is the reason that sparked the creation of several alternative low-rank definition and formats for tensors [27]). Nevertheless, a tensor analogue of the ACA procedure can still be formulated, as follows:

- A large enough pivot is determined in the tensor (the original one, or the current residual of the approximation)
- All the fibers[6] that contain the pivot are computed, and the unique rank-1 tensor $\hat{M}_1$ that coincides with M on these fibers is determined; M is replaced by $M - \hat{M}_1$, and the procedure is iterated.

Clearly, the "difficult" part is again the first item; two strategies have been experimented in this work:

1) A fixed number of entries is sampled from the tensor, and the element of maximum modulus is chosen as pivot.
2) Given the current fiber, the maximum along it is chosen as first pivot estimate. Then, another fiber through that pivot is chosen and the new candidate pivot is set as the maximum along the new fiber; the procedure is repeated a fixed number of times.

[6]The fibers for tensor play the role of rows and columns for matrices.

In the tests, procedure 2) has shown to be quite effective, and always converged in the considered tests. We remark that the underlying theory in this case is much weaker, and we currently expect that the proposed method may need to be adjusted in particular cases for $p \geq 2$. A possible modification is to restructure the method in a recursive way, where the tensor M is seen as $n_0 \dots n_l \times n_{l+1} \dots n_p$ with $l = \lceil \frac{p}{2} \rceil$, and the rows and columns in the ACA for this matrix are again approximated with an l-dimensional recursive ACA, with a base case at $p \leq 2$. This approach, described in [9], would be more expensive, but also more reliable. Nevertheless, the need for this more refined algorithm has never been encountered in the case studies considered in Section V.

D. Effective computation of parameters fiber

The use of a rational approximant to compute the entries of the j-th fiber still requires solving $k \cdot n_j$ linear systems with shifted versions of Q. This can easily become the bottleneck in the computation. Hence, we have modified the procedure to take advantage of the fact that these points are evaluations of a smooth function of a single parameter θ_j, as follows:

1) An interval $[\mathrm{lb}, \mathrm{ub}]$ enclosing all values of the parameter is determined.
2) The measure is evaluated at the Chebyshev points, defined as in Equation (7) of degree k (instead of degree n_j as used in Equation (7)).
3) The interpolant Chebyshev polynomial of degree $k-1$ is determined from these evaluations using a discrete cosine transform [20].
4) The measure is evaluated at new points, by setting $k := 2k-1$ (this only requires $k-1$ new evaluations, since the points are nested). If the previous approximant was accurate enough, the procedure is stopped, otherwise it is started again from item 3).

At the end of the above procedure, one has a high-quality approximant to the 1D function under consideration, which can be used to evaluate the function at all the required points.

The above algorithm makes the cost of the fiber evaluation almost independent of the number of points under consideration; the only part that depends on n_j is the final evaluation of the Chebyshev series. The number of solutions of linear systems with matrices $(z_h I + tQ)$, which are the dominating part in the cost, only depends on the smoothness of the evaluated function.

E. Evaluating accumulated measures

The algorithm described can be adapted to evaluate $m(t, \theta)$ when it is an accumulated measure, exploiting

$$\int_0^t \pi_0 e^{\tau Q} r \; d\tau = t\pi_0 \varphi_1(tQ) r = \mathrm{dot}(b, r),$$

where

$$\varphi_1(z) := \frac{e^z - 1}{z}, z \in \mathbb{C},$$

and noticing that the upper extreme of integration is t. The solution $b(t)$ of Equation (2) can then be expressed as $b(t) = t\pi_0\varphi_1(tQ)$. Using the rational approximation of e^z constructed from Gonchar poles one may derive a rational approximation for $\varphi_1(z)$ by computing

$$\hat{\xi}(z) := \frac{\xi(z) - 1}{z} = \sum_{h=1}^{k} \frac{\alpha_h z_h^{-1}}{z_h + z}.$$

This can be used in the analogous scheme to transform the evaluation of the fibers of M into a solution of a sequence of linear systems with shifted versions of Q, whose number is greatly reduced by exploiting the strategy of Section IV-D.

It can be noted that the accuracy attainable with this new approximation may be lower than the one for the matrix exponential, since one can only guarantee that

$$|z\varphi_1(z) - z\hat{\xi}(z)| \leq \epsilon,$$

which is a much looser bound for $|z| \ll 1$. To overcome this, the truncation threshold ϵ may be chosen as a smaller value. Alternatively, an optimal Remez approximation may be found for $\varphi_1(z)$ as well. In the experiments the slightly less optimal approximation $\hat{\xi}(z)$ is used for simplicity.

V. Case Studies

This section introduces two case studies, to illustrate the dependencies discussed in Section II-C and allow performance evaluation of the method presented in Section IV. Despite the apparent simplicity, the corresponding models include aspects that may lead to a variable size[7] and stiffness [4]–[6], and thus the proposed method is actually tested to compute $\hat{M}$ under unfavorable conditions.

A. Case study 1: degradable system

Consider a fault tolerant degradable system with n_r components inspired to that presented in [28]. The failure time of each component is exponentially distributed with parameter λ. The system is equipped with features to detect and recover from failures during normal system operation, which are successful with probability c_f (coverage parameter), leading to a degraded state where the failed component is recovered at a constant rate μ. Otherwise, with probability $1 - c_f$, the recovery of the component fails leaving the system in a state, labeled 0, in which all n_r components are temporarily non-operational, but the system has not yet failed, being still able to provide a basic service by default. The system is also equipped with features to detect and recover from state 0. Therefore it is recovered from state 0 with probability c_r at a constant rate μ_d. Otherwise, with probability $1 - c_r$, the overall system fails permanently (state 0_f): all components are considered no longer operational and no default degraded service can be delivered.

The CTMC representing this system is depicted in Figure 3, where each state $i \geq 0$ represents the number i of currently operational components.

The parameters defined in Case study 1 are $\theta_1 := \lambda$, $\theta_2 := c_f$, $\theta_3 := c_r$ and $\theta_4 := \mu_d$. Based on the number p of parameters actually considered 3 different sub-cases are defined:

[7] The model state-space is large.

Case study $1a$, Case study $1b$ and Case study $1c$, with p equal to 2, 3 and 4, respectively (and all the other parameters, beyond the first p, considered as constant numerical values).

The following two measures of interest have been analyzed:

Reliability: probability that the system is operational at time t given that it is operational at time 0, formally defined as $m(t,\theta) := \mathbb{E}[V(t,\theta)]$, where V is given in Equation (3) with reward structure

$$r_s = \begin{cases} 1 & \text{if } s \neq 0_f, \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

Under_repair: percentage of time during which at least one component is operational and at least one component is under repair. It is evaluated as the mean percentage of sojourn time the model spends in $n_r - 1, \ldots, 1$ in the interval $[0,t]$, i.e., time-averaged [3] accumulated measure formally defined by $m(t,\theta) := \mathbb{E}[Y(t,\theta)]/t$, where Y is given in Equation (4) with

$$r_s = \begin{cases} 1 & \text{if } 1 \leq s \leq n_r - 1, \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

B. Case study 2: System with UPS with limited battery life

Consider a system with a backup power generator based on a single battery with limited life time. The system is powered by a power supply unit which alternates between two states: Up, when power is supplied, and Down, when a power failure has occurred. The time to power failure and its duration (when the power supply is down) are exponentially distributed with rate λ and $\theta_1 := \mu$, respectively. Backup power generator is an Uninterruptible Power Supply (UPS) unit which provides instant continuation of electrical current after a power failure. The system fails as soon as the power supply is down and the battery of the UPS is exhausted. During a power failure, the UPS takes over until regular power is restored or until the battery is exhausted, when the system fails. The battery rundown time, during a power failure, is a deterministic time with parameter $\theta_2 := L$. It is assumed that, when the power is restored, the battery will be fully recharged before the next power failure occurs. The life time of the battery depends on the number of discharging events of the battery for more than 85% of its charge (destructive discharge). When the number of such destructive discharge cycles has reached a certain threshold n_c, then the battery can no longer be recharged.

The semi-Markov chain model representing this system is depicted in Figure 5. Different states are modeled depending on the two battery charge levels B considered during a power failure: "$B > 15\%$" or "$0\% < B < 15\%$", where the battery rundown times are deterministic with parameter $0.85 \cdot L$ and $0.15 \cdot L$, respectively. Moreover, discharging the battery for more than 85% of its charge moves the model into a state where the number of battery destructive discharge cycles completed is increased by 1. This situation is represented, for example, by the transition from state $(0, Down, B > 15\%)$ to $(1, Down, 0 < B < 15\%)$ (for the state notation see caption of Figure 5).

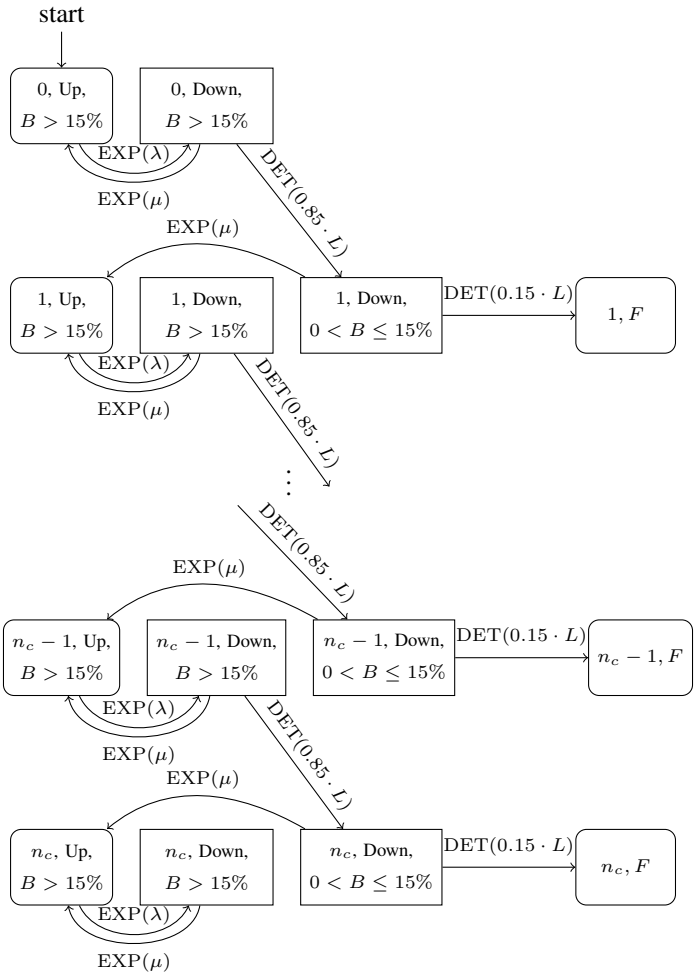


Fig. 5. Case study 2: Semi-Markov chain for system with UPS with limited battery life. Rounded and non-rounded rectangles represent states with exponential and non-exponential respectively sojourn times. Each transient state is a tuple that contains: the number of battery destructive discharge cycles completed, the state of the power supply (up or down), and the battery charge level (greater than 15% or between 0 and 15%). Absorbing states are labeled with: life cycle number and F (*system failure*). Each arc is labeled with the probability distribution from which the transition time is drawn.

In order to solve the model, it is converted into a CTMC, approximating the $[100\%, 85\%)$ and $(15\%, 0\%)$ discharge times with Erlang distributions comprising $n_{e,1}$ and $n_{e,2}$ exponential transitions, respectively.

The following two measures of interest have been analyzed:

Reliability: defined as $m(t,\theta) := \mathbb{E}[V(t,\theta)]$, where V is given in Equation (3) with reward

$$r_s = \begin{cases} 0 & \text{if } s = (i, \mathrm{F}), \text{ with } i = 1, \ldots, n_c, \\ 1 & \text{otherwise.} \end{cases} \tag{12}$$

Low_charge: percentage of time where the UPS works with low battery, formally defined as $m(t,\theta) := \mathbb{E}[Y(t,\theta)]/t$,

where Y is given in Equation (4) with

$$r_s = \begin{cases} 1 & \text{if } s = (i, \text{Down}, 0 < B \leq 15\%), \\ & \text{with } i = 1, \ldots, n_c, \\ 0 & \text{otherwise.} \end{cases} \quad (13)$$

VI. Performance Evaluation

The aim of this section is to compare the performance of the approach presented in Section IV. Two alternatives are considered for the computation of the solution bundle: the Full Tensor and the Generalized Linear Model (GLM). To concretely show an example of the computed measures bundle, results of sensitivity analysis applied to selected measures of the two case studies are briefly shown.

A. Comparison with the full tensor evaluation alternative

The comparison with the full tensor M evaluation when high accuracy of the measures is required focuses on the definition of $\hat{M}$ as in Equation (8) (that is the precursor to obtain $\hat{m}$ as Equation (6)).

In all the experiments, the mission time is set to 10 years, corresponding to $t_{\max} = 24 \cdot 365 \cdot 10$ hours, and the ACA tolerance is set to 10^{-6}. This, together with parameters assignments, leads to stiff models [11]. The performance of the approach as the state-space dimension N increases is not reported, even though it has been verified that, as expected, the speedups reported in Tables I and II remain almost the same at increasing of N. To summarize the results: speedups of the proposed approach with respect to the full evaluation of the tensor in both case studies are relevant, and grow at increasing of points number $n_0 \cdot n_1 \cdot n_2$, maintaining great accuracy.

The computer where the experiments were performed has a 11^{th} Gen Intel(R) Core(TM) i7-1165G7 CPU, 4 unit clocked at 2.80GHz, 8 treads, 40Gb of DDR4 RAM clocked at 3200MHz. The operating system is Pop!_OS 20.04 LTS. The implementation of the solution method has been written in MATLAB [29] and is freely available[8]. Memory consumption for each run of Case 1 and 2 has been between 4 and 6 Gb.

The parameters are those detailed in Section V. The parameters' assignment (rates are expressed in hour^{-1}) for Case study 1a is: $\mu = \mu_d = 0.5$, $c_r = 0.9, \theta_1 := \lambda \in [10^{-6}, 10^{-5}]$, $\theta_2 := c_f \in [0.9, 0.99]$. For Case study 1$b$, $\theta_3 := c_r \in [0.9, 0.99]$, and, for Case study 1$c$, $\theta_4 := \mu_d \in [0.25, 0.75]$. Speedup and accuracy are reported in Table I for Reliability and Under_repair, where it can be appreciated that the proposed approach requires few seconds to about 10^6 points (more precisely: $n_0 \cdot n_1 \cdot n_2 = 100^3$, for Case study 1$a$; $n_0 \cdot n_1 \cdot n_2 \cdot n_3 = 32^4$, for Case study 1$b$; and $n_0 \cdot n_1 \cdot n_2 \cdot n_3 \cdot n_4 = 16^5$, for Case study 1$c$), whereas the full evaluation of M require hundreds of seconds, and the accuracy (obtained comparing the two) is close to or better than 10^{-6}. It is also interesting to notice that for the approximation $\hat{M}$ a really small fraction of points (k instead of 10^6) are needed, and that the adaptive choice of pivots described in Section IV-C determines a quick decrease of pivots' magnitude, as depicted in Figure 6.

[8]https://github.com/106ohm/MarkovCrossApproximation

TABLE I
CASE1: WALL-CLOCK TIME (IN SECONDS) REQUIRED TO COMPUTE THE FULL TENSOR AND THE ACA APPROXIMATION, SPEEDUP AND ACCURACY. HERE $N = 4$ AND ABOUT 10^6 EVALUATION POINTS.

	Case study 1a ($p = 2$)		Case study 1b ($p = 3$)	Case study 1c ($p = 4$)
	Under_repair	Reliability	Reliability	Reliability
k	6	6	11	15
M (seconds)	$2.86 \cdot 10^2$	$2.31 \cdot 10^2$	$7.91 \cdot 10^2$	$1.76 \cdot 10^3$
$\hat{M}$ (seconds)	0.77	0.67	0.91	1.39
accuracy	$3.79 \cdot 10^{-7}$	$1.61 \cdot 10^{-6}$	$6.96 \cdot 10^{-7}$	$5.80 \cdot 10^{-7}$
speedup	371	344	874	1263

TABLE II
CASE 2: WALL-CLOCK TIME (IN SECONDS) REQUIRED TO COMPUTE THE FULL TENSOR AND THE ACA APPROXIMATION, SPEEDUP AND ACCURACY.

	Reliability	Reliability	Reliability	Low_charge
N	12	12	12	12
$n_0 n_1 n_2$	10^3	$8 \cdot 10^3$	10^5	10^6
k	30	32	34	27
M	$1.17 \cdot 10^2$	$4.48 \cdot 10^2$	$9.812 \cdot 10^3$	$1.36 \cdot 10^4$
$\hat{M}$	$9.7 \cdot 10^1$	$9.5 \cdot 10^1$	$8.55 \cdot 10^1$	$1.84 \cdot 10^2$
accuracy	$2.63 \cdot 10^{-6}$	$5.08 \cdot 10^{-7}$	$7.56 \cdot 10^{-7}$	$8.87 \cdot 10^{-6}$
speedup	1.2	4.69	115	74

For Case study 2, parameters' assignment (rates are expressed in hour^{-1}) is:

$$n_c = 3,\ \lambda = 10^{-5},\ n_{e,1} = 5,\ n_{e,2} = 3,$$
$$\theta_1 := L \in [1, 3],\ \theta_2 := \mu \in [0.5, 2.5].$$

Speedup and accuracy are reported in Table II for Reliability at increasing of points number (specifically, n_0 is kept equals to 10, and both n_1 and n_2 span $\{10, 20, 100\}$), and for Low_charge with 10^6 points. As expected, the speedup increases at the increase of points number. Indeed, k remains low (about 30) but parameters' fibers require an increasing number of approximation points. Notice that both the strict definition of accuracy employed here and the adaptive choice of pivots are responsible for obtaining a smaller accuracy with 32 and 34 approximation points than with 32.

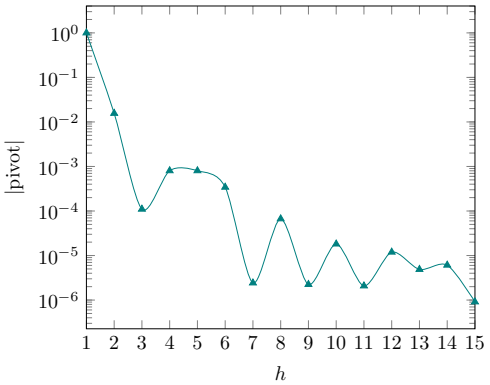


Fig. 6. Case study 1c: Absolute value of the pivot when computing Reliability as a function of the steps $h = 1, \ldots, k$. In this run ACA stops at $k = 15$.

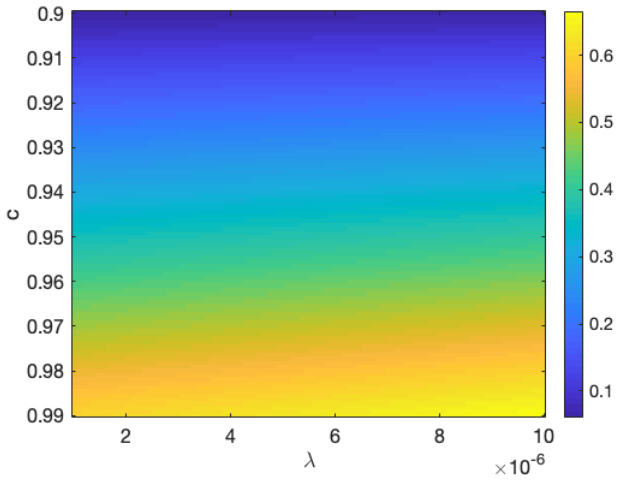


Fig. 7. Case study $1a$: Under_repair for $t_{\max} = 2$ years.

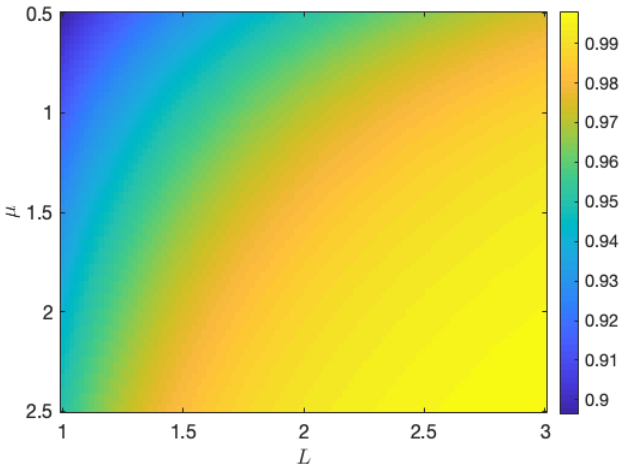


Fig. 8. Case study 2: Reliability for $t_{\max} = 2$ years.

B. Comparison with GLM

To enrich the comparison study, the GLM metamodel has been selected as another alternative to the proposed ACA approximation. Generalized Linear Model (GLM), here defined consistently with the rest of the paper as

$$\hat{m} = \mathbb{E}[m|t,\theta] = g^{-1}\big(\beta_{-1}+\beta_0 \cdot t+\beta_1 \cdot h_1(\theta_1)+\cdots+\beta_p \cdot h_p(\theta_p)\big),$$

is the ancestor of many interpretable metamodels [10]. Hence, choosing GLM not only allows to perform a quantitative comparison, but also to discuss qualitative differences between ACA and descendants of GLM. In particular:

- in GLM a case-by-case choice of dependent variable distribution (within the exponential family) and link function g is needed because Equations (3) and (4) can have different ranges and shapes, whereas ACA is general;
- the independent variable functions h_i are also chosen case-by-case, taking into account the discussion of Section II-C and how it is expected that a given parameter influences the final measure;
- the parameters vector β in GLM has always $p+2$ entries, whereas in ACA the number of approximants k is chosen dynamically. Thus, GLM is not expected to gain much in accuracy at increasing of points number over a threshold of the order of p.

Numerical evaluations have been carried out on Reliability in Case study 1 for $n_0 n_1 n_2 = 8 \cdot 10^3$, choosing the Gamma family with default link function and exploiting MATLAB `fitglm`. Several arranges of points and functions of the independent variables have been considered. The best relative accuracy reached training the GLM metamodel is $1.40 \cdot 10^{-2}$, obtained with 100 randomly distributed points (with 10 points the accuracy is $2.21 \cdot 10^{-2}$). With the approach of Section IV, $k = 9$ is sufficient to obtain a relative accuracy of $3.19 \cdot 10^{-7}$. The higher accuracy of the ACA approximation is payed by a higher computation time: GLM is about 37 times faster than ACA. However, it has to be considered that both require less than one second to perform the approximation (while the full tensor evaluation requires about 9 seconds).

In `fitglm` it is also possible to introduce mixed terms in the approximation, but loosing a little bit of interpretability. Through trial and error, the best accuracy obtained has been $2.59 \cdot 10^{-4}$ with the option `poly233`, and almost the same training time.

C. Results from sensitivity analysis

To exemplify the utility of the computed measure bundle, global sensitivity analysis has been selected among the kind of analyses that greatly benefit from the presented contribution, as discussed in Section II-B.

Specifically, the measure Under_repair of Case study $1a$ and the measure Reliability of Case study 2 have been considered. Heatmaps of the approximated measures $\hat{m}$, for $t_{\max}$ equals to 2 years and at varying parameter values, are shown in Figures 7 and 8 to point out different parameters' interplay. Figure 7 confirms that c is more relevant than λ for Under_repair, and the increase/decrease pattern follows the axes directions. In Figure 8, instead, bend curves of equal Reliability appear, meaning that more complex design choices are possible.

VII. Related Work

Narrowing the discussion to aspects directly related to the contribution, i.e., find ways to avoid the full measures tensor evaluation, a first observation is that the literature focuses mainly on the time fiber. Integration- and simulation- based approaches differ from the expansion- and approximation-based ones for a relevant aspect. In fact, the former impose a causal order among time points, so the computation of the model solution cannot be parallelized in time. Instead, uniformization and semi-symbolic [11], [30], [31] just work with a single time point (trivial case of no time causal order), and approximation as the one presented here can be done with a high level of parallelism, even exploiting dedicated hardware. Indeed, importing in the approach of Section IV a different

method to evaluate the tensor along time fiber, e.g. [7], does not impact on the presented speedup.

The main issue, as briefly illustrated in Section I, with the previously mentioned strategies is that, even if they are efficient/effective in solving one system of ODEs, dealing with large values of n_{sample} can quickly become unfeasible. Among the strategies to mitigate this issue that have been already published, in the following only those that explicitly address the parameters fiber are discussed.

In [16], [17] a metamodel (also known as surrogate model or emulator in the literature) is presented. The performability model is solved on a predefined set of time and parameters points, the dataset $\{(t^{(h)}, \theta^{(h)}), m_{\theta^{(h)}}(t^{(h)})\}_h$ is divided in training/test/validation and $\hat{m}(t, \theta)$ is learned. The results show that the required training set size is much smaller than the number of points of the full tensor M, as also in this paper. However, how to choose the samples (for instance through random sampling, Latin hypercube sampling, and Sobol sequence sampling) can be challenging. The training set is fixed at the beginning, whereas the approach presented in this paper is adaptive. Comparing the impact of this difference on accuracy is beyond the scope of this paper. No assumption is made on measure smoothness, and actually the performability model can be solved through simulation, thus the approach addresses models more general than Markov. Instead of adopting a single approximation/machine learning method, the authors present a stack of methods to enhance accuracy with respect to each method exercised in isolation: a group of methods work on input data, their results are compared and fed to a second group of methods, and so on. It follows that, even though some of the methods can be interpretable, the stack $\hat{m}$ is trained as a black box, so interpretability is lost. Further work is required to investigate how well-known or new explainability techniques [10] can increase trust in this approach.

Equations (1) and (2) are systems of PDEs (even though they can be actually seen as fake systems of ODEs because involve only partial derivative on time), thus natural competitors of the proposed approach can be searched among approximation/machine learning approaches addressing PDEs. A strategy for computing the solution bundle has been recently presented in [8], where the case studies have a small number of dependent variables, but closed form solutions, if available, are too complex to be exploitable in practice. A nonlinear, non separable, approximation is chosen (approximants are Deep Neural Networks), so this is a black box approach too. When dealing with moderate and large values of N, the strategy of [8] is no more applicable in the context of Markov models, because automatic differentiation is too expensive (actually this issue can be mitigated by exploiting finite difference) and relevant properties of the solution bundle, such as for Equation (1) being a state probability vector, are not addressed. Preliminary investigations show that accuracy can be also problematic, since performability models typically require high accuracy (e.g., when targeting critical systems), well beyond values reported in [8] related to other contexts. Whether or not it is appropriate to address performability models in this way requires further investigations.

Weakening the usual definitions of surrogate model and solution bundle, the approach presented in [32], dealing with Continuous Time Continuous Space Kolmogorov PDE, could be relevant too.

VIII. Conclusions and future work

In this paper a new low rank approximation $\hat{M}$ of performability measures, based on the ACA technique, has been presented for Markov models. The main challenge, i.e., how to address parameters' fibers to gain in efficiency with respect to the full evaluation of the tensor M, has been addressed tailoring well-known results to the measures of interest in Sections IV-D and IV-E. The resulting approximation approach is accurate (as shown in Section VI) and trustworthy because of the decomposability property promoted by exploiting separable approximants.

Future work is foreseen in several directions. As already discussed, the presented approach belongs to the category of interpretable solutions. Comparison with existing alternatives in the explainable (black box) category (such as [16], [17]) would be an interesting investigation to carry on. This research requires careful identification of the baseline and perspectives for the common comparison.

It is known that ACA performs really well only if the measure under analysis is sufficiently smooth in time and parameters. Thus, tweaking the presented approach to address models that are more general than Markov, maintaining a good performance, is doable as long as the resulting measures are smooth. For instance, the presented approach is ready to address instantaneous measures on non-homogeneous CTMC [1], but, to obtain good speedups with respect to the full evaluation of M, the developments of Section IV-E has to be improved.

Semi-Markov models, under not-so-restrictive conditions on the involved Cumulative Distribution Functions, are clearly the next kind of models to address. In addition, case study 2 is actually formulated as a semi-Markov model, and then it would be interesting to compare the reported performance, obtained applying the presented approach on a Markovian approximation of this case study, with that obtainable directly addressing the semi-Markov model. Of course, the challenge here is how to adapt solutions presented in Sections IV-D and IV-E to the semi-Markov case.

Generalizations in other directions are foreseen as well. For instance, considering a time- and parameters- dependent reward $r(t, \theta)$ poses no issues to the approximation of instantaneous measures, as long as it is a smooth function, but again ask for additional work when addressing accumulated measures.

In the literature, it is also known that ACA starts to face troubles when the number of parameters p becomes large. Thus, to the best of the authors' knowledge, the best choice to address several parameters is to adopt one of the approaches presented in [16], [17]. Nonetheless, generalizations of ACA to address high dimensional tensors exist in the literature, and then the next step is to apply and test those generalizations to the presented approach.

REFERENCES

[1] K. S. Trivedi and A. Bobbio, *Reliability and Availability Engineering: Modeling, Analysis, and Applications*, 2017.

[2] B. R. Haverkort and K. S. Trivedi, "Specification techniques for Markov reward models," *Discrete Event Dyn. Syst.*, no. 3, pp. 219–247, 1993.

[3] W. H. Sanders and J. F. Meyer, "A unified approach for specifying measures of performance, dependability and performability," *Dependable Computing for Critical Applications, Vol. 4 of Dependable Computing and Fault-Tolerant Systems*, pp. 215–237, 1991.

[4] A. L. Reibman and K. S. Trivedi, "Transient analysis of cumulative measures of Markov model behavior," *Communications in Statistics. Stochastic Models*, vol. 5, no. 4, pp. 683–710, 1989.

[5] A. L. Reibman, R. Smith, and K. S. Trivedi, "Markov and Markov reward model transient analysis: An overview of numerical approaches," *European Journal of Operational Research*, vol. 40, pp. 257–267, 1989.

[6] M. Malhotra, J. K. Muppala, and K. S. Trivedi, "Stiffness-tolerant methods for transient analysis of stiff Markov chains," *Microelectronics Reliability*, vol. 34, no. 11, pp. 1825–1841, 1994.

[7] A. V. Ramesh and K. Trivedi, "Semi-numerical transient analysis of Markov models," in *Proceedings of the 33rd Annual on Southeast Regional Conference*, ser. ACM-SE 33, 1995, pp. 13–23.

[8] C. W. Flamant, P. Protopapas, and D. Sondak, "Solving differential equations using neural network solution bundles," *ArXiv*, vol. abs/2006.14372, 2020.

[9] M. Bebendorf, "Adaptive cross approximation of multivariate functions," *Const. Approx.*, vol. 34, no. 2, pp. 149–179, 2011.

[10] R. Marcinkevičs and J. E. Vogt, "Interpretability and explainability: A machine learning zoo mini-tour," *ArXiv*, vol. 2012.01805, 2020.

[11] M. Malhotra, J. K. Muppala, and K. S. Trivedi, "Stiffness-tolerant methods for transient analysis of stiff Markov chains," *Microelectronics Reliability*, vol. 34, no. 11, pp. 1825–1841, 1994.

[12] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward, "SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers," *ACM Transactions on Mathematical Software (TOMS)*, vol. 31, no. 3, pp. 363–396, 2005.

[13] R. L. Iman and J. C. Helton, "An investigation of uncertainty and sensitivity analysis techniques for computer models," *Risk Analysis*, vol. 8, no. 1, pp. 71–90, 1988.

[14] É. Walter and L. Pronzato, *Identification of parametric models: from experimental data*, ser. Communications and Control Engineering. Heidelberg: Springer-Verlag, 1997.

[15] H. Pham, *Reliability Modeling, Analysis and Optimization*, ser. on Quality, Reliability and Engineering Statistics, M. Xie, T. Bendell, and A. P. Basu, Eds. Singapore: World Scientific, 2006, vol. 9.

[16] M. Rausch and W. H. Sanders, "Sensitivity analysis and uncertainty quantification of state-based discrete-event simulation models through a stacked ensemble of metamodels," in *Quantitative Evaluation of Systems*, 2020, pp. 276–293.

[17] ——, "Evaluating the effectiveness of metamodeling in emulating quantitative models," in *Quantitative Evaluation of Systems*, 2021, pp. 127–145.

[18] A. Townsend and L. N. Trefethen, "An extension of Chebfun to two dimensions," *SIAM J. Sci. Comput.*, vol. 35, no. 6, pp. C495–C518, 2013. [Online]. Available: https://doi.org/10.1137/130908002

[19] R. A. Maire, A. L. Reibman, and K. S. Trivedi, "Transient analysis of acyclic Markov chains," *Performance Evaluation*, vol. 7, no. 3, pp. 175–194, 1987.

[20] L. N. Trefethen, *Approximation theory and approximation practice*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2013.

[21] A. J. Carpenter, A. Ruttan, and R. S. Varga, "Extended numerical computations on the 1/9 conjecture in rational approximation theory," in *Rational approximation and interpolation*. Springer, 1984, pp. 383–411.

[22] B. Hashemi and L. N. Trefethen, "Chebfun in three dimensions," *SIAM J. Sci. Comput.*, vol. 39, no. 5, pp. C341–C363, 2017. [Online]. Available: https://doi.org/10.1137/16M1083803

[23] J. W. Demmel, *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997. [Online]. Available: https://doi.org/10.1137/1.9781611971446

[24] T. F. Chan, "Rank revealing QR factorizations," *Linear Algebra Appl.*, vol. 88/89, pp. 67–82, 1987. [Online]. Available: https://doi.org/10.1016/0024-3795(87)90103-0

[25] N. Halko, P. G. Martinsson, and J. A. Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011. [Online]. Available: https://doi.org/10.1137/090771806

[26] M. Bebendorf, "Approximation of boundary element matrices," *Numer. Math.*, vol. 86, no. 4, pp. 565–589, 2000. [Online]. Available: https://doi.org/10.1007/PL00005410

[27] L. Grasedyck, D. Kressner, and C. Tobler, "A literature survey of low-rank tensor approximation techniques," *GAMM-Mitt.*, vol. 36, no. 1, pp. 53–78, 2013. [Online]. Available: https://doi.org/10.1002/gamm.201310004

[28] J. B. Dugan and K. S. Trivedi, "Coverage modeling for dependability analysis of fault-tolerant systems," *IEEE Trans. on Comp.*, vol. 38, no. 6, pp. 775–787, 1989.

[29] MathWorks, *MATLAB R2018a*, The Mathworks, Inc., 2018.

[30] A. L. Reibman, R. Smith, and K. S. Trivedi, "Markov and Markov reward model transient analysis: An overview of numerical approaches," *European Journal of Operational Research*, vol. 40, no. 2, pp. 257–267, 1989.

[31] A. Reibman and K. S. Trivedi, "Transient analysis of cumulative measures of Markov model behavior," *Communications in Statistics. Stochastic Models*, vol. 5, no. 4, pp. 683–710, 1989.

[32] J. Berner, M. Dablander, and P. Grohs, "Numerically solving parametric families of high-dimensional Kolmogorov partial differential equations via deep learning," *ArXiv*, vol. abs/2011.04602, 2020.