The Arnoldi Aggregation for Approximate Transient Distributions of Markov Chains

Patrick Sonnentag(✉)[0009−0002−7319−5884], Fabian Michel[0009−0005−7768−9111], and Markus Siegle[0000−0001−7639−2280]

University of the Bundeswehr Munich, Institute for Computer Engineering, Werner-Heisenberg-Weg 39, Neubiberg, 85579, Germany
{patrick.sonnentag,fabian.michel,markus.siegle}@unibw.de

Abstract The paper proposes a new aggregation method, based on the Arnoldi iteration, for computing approximate transient distributions of Markov chains. This aggregation is not partition-based, which means that an aggregate state may represent any portion of any original state, leading to a reduced system which is not a Markov chain. Results on exactness (in case the algorithm finds an invariant Krylov subspace) and minimality of the size of the Arnoldi aggregation are proven. For practical use, a heuristic is proposed for deciding when to stop expanding the state space once a certain accuracy has been reached. Apart from the theory, the paper also includes an extensive empirical section where the new aggregation algorithm is tested on several models and compared to a lumping-based state space reduction scheme.

Keywords: Aggregation · Arnoldi Iteration · Krylov Subspace Method · Markov Chains · State Space Reduction

1 Introduction & Motivation

Approximate solutions of Markov chains by means of state space reduction have received a lot of attention in the literature. One of the earliest works is that of Simon and Ando [19] on nearly decomposable Markov chains. These techniques were further developed and applied to models of computer systems by Courtois [8] which also led to iterative aggregation/disaggregation methods [13]. Most of these early works on state space aggregation focused on the stationary distribution, while Buchholz in his seminal paper on lumpability [6] also considered transient distributions. An overview of general transient solution methods is also given in the paper [18], where — among several others — a Krylov subspace based method is briefly described. It is, of course, important to control the error caused by aggregation/disaggregation; see, for instance, [7]. Recently, state space reduction techniques for obtaining approximate transient distributions of Markov chains with formal error bounds have been studied [1]. This has been generalized in [16], the latter work characterising exactness of Markov chain aggregations, which goes beyond the well-known concept of lumping.

To achieve a state space reduction of a Markov chain with transition matrix P, the primary question is how to aggregate the state space, i.e. how to construct the reduced system. More concretely, the question is how to choose the size and the transition matrix of the reduced system. Secondly, once a solution for the reduced system has been obtained, a strategy is needed for mapping this solution back onto the original chain, i.e. one needs to specify a disaggregation scheme, usually in the form of a disaggregation matrix A. The third question concerns the choice of the initial vector for the aggregated system.

Common approaches to state space reduction use a partitioning of the original state space, leading to one aggregated state per class, which means that every original state belongs to exactly one state of the reduced system. The reduced system is usually considered to be a Markov chain again, its transition matrix constructed as a stochastic matrix, and the initial vector of the reduced system a probability vector. In this paper, we consider more general state space reductions, where a reduced state can represent any portion of any original state.

The Arnoldi iteration [2] is a well-known algorithm for computing an orthonormal basis of the Krylov subspace of a vector v w.r.t. a matrix M, i.e. the space span $\left\{ v^\intercal, v^\intercal M, \ldots, v^\intercal M^{j-1} \right\}$. We develop the Arnoldi aggregation for Markov chains, based on the Arnoldi iteration, where the initial vector is the initial distribution ($v = p_0$) and $M = P$ is the transition matrix of the DTMC. The vectors computed as the basis of this Krylov subspace are used as the rows of the disaggregation matrix A, and the step matrix of the reduced system is obtained from the multiplication factors appearing in the Gram-Schmidt orthonormalization part of the Arnoldi process. The step matrix thus obtained is not a stochastic matrix, i.e. the reduced system is not a Markov chain, and — in consequence — the vectors obtained may not be probability distributions. We show that Arnoldi aggregations are either exact with minimal state space size or have an error of zero for the first $j - 1$ time steps (where j is the number of Arnoldi iterations performed) with minimal state space size. For transient time points larger than the number of Arnoldi iterations (which is the interesting case), the error can be estimated by a heuristic during the Arnoldi iteration, thus making it possible to dynamically control the size of the reduced system based on the desired error.

The key contributions of this paper, which is based on the thesis [20], are as follows: A new aggregation scheme for Markov chains is proposed and analysed from a theoretical point of view, leading to exactness and minimality results. In view of finite floating point precision, practically viable termination criteria for the iterative Arnoldi procedure in the aggregation setting are established. Empirical results are provided which confirm the theoretical findings and show that the Arnoldi aggregation can be superior to other approaches.

The remainder of the paper is structured as follows: Starting in Section 2, we explain basic notation and formally introduce the Arnoldi iteration. In Section 3, we introduce the Arnoldi aggregation and derive its main theoretical properties. We further consider how to work with Arnoldi aggregations in floating point arithmetic, introduce the final algorithm and analyse its runtime and memory complexity. Lastly, in Section 4, Arnoldi aggregations are evaluated and compared

to Exlump aggregations from [16, Algorithm 3] on a range of different models. A summary and outlook for future work are given in Section 5.

2 Preliminaries

2.1 Notation

Let $u, v \in \mathbb{R}^n$ and $M \in \mathbb{R}^{n \times m}$. Then, $|v|$ and $|M|$ denote the vector and the matrix with the absolute value applied component-wise. Furthermore, denote with $\mathbf{0}_n$ and $\mathbf{1}_n$ the vectors in $\mathbb{R}^n$ with all entries zero or one, respectively. Let $e_j \in \mathbb{R}^j$ be the jth standard basis vector of $\mathbb{R}^j$, and $I_n \in \mathbb{R}^{n \times n}$ the identity matrix. Lastly, define $\langle u, v \rangle \coloneqq u^\intercal v$, $\|v\|_1 \coloneqq \langle |v|, \mathbf{1}_n \rangle$ and the matrix norm $\|M\|_\infty$ as the maximum absolute row sum of M.

Here, we are going to work with time-homogeneous discrete-time Markov chains with finite state spaces $S = \{\, 1, \ldots, n \,\}$ solely. Furthermore, if X_k is the state of the Markov chain at time k, let $P \in \mathbb{R}^{n \times n}$ be the transition matrix of the Markov chain where $P(i, j) \coloneqq \mathbb{P}(X_{k+1} = j \mid X_k = i)$. With an initial distribution $p_0 \in \mathbb{R}^n$, transient distributions are given by $p_k^\intercal = p_0^\intercal P^k$.

Next, we consider state space reductions through aggregations. In an aggregation of dimension $m \leq n$, the (arbitrary) *aggregated step matrix* $\Pi \in \mathbb{R}^{m \times m}$ shall approximate the dynamics of P. Let $\pi_0 \in \mathbb{R}^m$ be the (arbitrary) *initial aggregated vector* and $\pi_k^\intercal \coloneqq \pi_0^\intercal \Pi^k$ the *transient aggregated vector*. To return to an n-dimensional space, we use the (arbitrary) *disaggregation matrix* $A \in \mathbb{R}^{m \times n}$ to define *approximated transient distributions* as $\tilde{p}_k^\intercal \coloneqq \pi_k^\intercal A$. Finally, following [16, Def. 8], an aggregation is *dynamic-exact* if $\Pi A = AP$. If further $p_0 = \tilde{p}_0$ holds, it is *exact*. This is motivated by dynamic-exactness implying $\tilde{p}_k^\intercal = \tilde{p}_0^\intercal P^k$ and exactness even offering $p_k = \tilde{p}_k$. The kth *transient error vector* is given through $\mathrm{err}_k \coloneqq \tilde{p}_k - p_k$ with $\|\mathrm{err}_k\|_1$ denoting the *transient error* after k steps.

2.2 Arnoldi Iteration

For a vector $v \in \mathbb{R}^n$ and a matrix $M \in \mathbb{R}^{n \times n}$, the Arnoldi iteration, as first introduced in [2], builds an orthonormal basis $q_1^\intercal, \ldots, q_j^\intercal$ of the Krylov subspace

$$\mathcal{K}^j(v, M) \coloneqq \operatorname{span} \left\{\, v^\intercal, v^\intercal M, \ldots, v^\intercal M^{j-1} \,\right\},$$

using the Gram-Schmidt procedure, as done in Algorithm 1. There, we can see the Gram-Schmidt procedure in lines 2, 4–9 and 12; lines 3, 10 and 13 construct some matrices $H_j \in \mathbb{R}^{j \times j}$ and $Q_j \in \mathbb{R}^{j \times n}$, and line 11 provides the return value once some arbitrary termination criterion is met (suggestions for this will follow later). Usually, $h_{j,j+1} = 0$ is part of this criterion to avoid division by zero. The relation

$$H_j Q_j + h_{j,j+1} e_j q_{j+1}^\intercal = Q_j M \tag{1}$$

holds if $\mathcal{K}^j(v, M) \subsetneq \mathcal{K}^{j+1}(v, M)$ (with $e_j \in \mathbb{R}^j$). If $\mathcal{K}^j(v, M) = \mathcal{K}^{j+1}(v, M)$ holds, (i.e., the Krylov subspace is invariant under M), Eq. (1) reduces to

$$H_j Q_j = Q_j M. \tag{2}$$

Algorithm 1 Arnoldi Iteration

Let $v \in \mathbb{R}^n$, $M \in \mathbb{R}^{n \times n}$.
$q_1 \coloneqq \frac{v}{\|v\|_2}$
$Q_1 \coloneqq \begin{pmatrix} — & q_1^\intercal & — \end{pmatrix}$
for $j = 1, 2, \ldots$ **do**
  $r_1^\intercal \coloneqq q_j^\intercal M$
  for all $i = 1, 2, \ldots, j$ **do**
    $h_{j,i} \coloneqq \langle r_i, q_i \rangle$
    $r_{i+1} \coloneqq r_i - h_{j,i} q_i$
  $h_{j,j+1} \coloneqq \|r_{j+1}\|_2$
  $H_j \coloneqq \begin{cases} \begin{pmatrix} h_{1,1} \end{pmatrix} & \text{if } j = 1 \\ \begin{pmatrix} H_{j-1} & \mathbf{0}_{j-2} \\ & h_{j-1,j} \\ h_{j,1} \cdots & h_{j,j} \end{pmatrix} & \text{else} \end{cases}$
  if criterion is true **then return** H_j, Q_j
  $q_{j+1} \coloneqq \frac{r_{j+1}}{h_{j,j+1}}$
  $Q_{j+1} \coloneqq \begin{pmatrix} Q_j \\ — & q_{j+1}^\intercal & — \end{pmatrix}$

While not hard to show, proving Eqs. (1) and (2) thoroughly requires lengthy calculations. Refer to [17, Sec. 6.2] for a proof. There, a slight variation of the Arnoldi iteration is used, with rearranged order of operations for the underlying Gram-Schmidt procedure.

3 Using the Arnoldi Iteration to Build Aggregations

Both Eqs. (1) and (2), but especially Eq. (2), are reminiscent of the definition of dynamic-exact aggregations of Markov chains in Section 2.1. Using these as a basis while further guaranteeing $p_0 = \tilde{p}_0$ motivates:

Definition 1 (Arnoldi aggregation). *With $P \in \mathbb{R}^{n \times n}$ as the transition matrix of a Markov chain and $p_0 \in \mathbb{R}^n$ an initial distribution, the aggregation with $\Pi \coloneqq H_j$, $A \coloneqq Q_j$, resulting from the Arnoldi iteration with P and p_0, and $\pi_0 \coloneqq (\|p_0\|_2, 0, \ldots, 0)^\intercal \in \mathbb{R}^j$, is the* Arnoldi aggregation *of size j for P and p_0.*

3.1 Exact and Initially Exact Arnoldi Aggregations

We start by exploring some properties of Arnoldi aggregations and their connection to (initially) exact aggregations.

Lemma 1. *In an Arnoldi aggregation of size j, it holds that*

$$\pi_k = (\underbrace{\lambda_1, \ldots, \lambda_{k+1}}_{all \in \mathbb{R}}, \underbrace{0, \ldots, 0}_{(j-k-1)\times})^\intercal \quad \text{for } 1 \leq k \leq j-1 \text{ and for some real numbers } \lambda_i.$$

Proof. Follows by induction on k and the form of H_j.

Lemma 2. *In Arnoldi aggregations of size j after k steps, we have*

$$\pi_0^\intercal H_j^k Q_j + \sum_{i=0}^{k-1} \pi_0^\intercal H_j^i \left(h_{j,j+1} e_j q_{j+1}^\intercal\right) P^{k-1-i} = \pi_0^\intercal Q_j P^k$$

Proof. By induction on k. Clearly, $\pi_0^\intercal I_j Q_j + 0 = \pi_0^\intercal Q_j I_n$. Induction step:

$$\begin{aligned}
\pi_0^\intercal Q_j P^{k+1} &= \pi_0^\intercal H_j^k Q_j P + \sum_{i=0}^{k-1} \pi_0^\intercal H_j^i \left(h_{j,j+1} e_j q_{j+1}^\intercal\right) P^{k-i} \\
&\overset{(1)}{=} \pi_0^\intercal H_j^k \left(H_j Q_j + h_{j,j+1} e_j q_{j+1}^\intercal\right) + \sum_{i=0}^{k-1} \pi_0^\intercal H_j^i \left(h_{j,j+1} e_j q_{j+1}^\intercal\right) P^{k-i} \\
&= \pi_0^\intercal H_j^{k+1} Q_j + \sum_{i=0}^{k} \pi_0^\intercal H_j^i \left(h_{j,j+1} e_j q_{j+1}^\intercal\right) P^{k-i}.
\end{aligned}$$

Proposition 1. *An Arnoldi aggregation of size j has $p_k = \tilde{p}_k$ for $0 \le k \le j-1$.*

Proof. Combine Lemmas 1 and 2 to get

$$\begin{aligned}
p_k^\intercal = \pi_0^\intercal Q_j P^k &\overset{\text{Lem. 2}}{=} \pi_0^\intercal H_j^k Q_j + \sum_{i=0}^{k-1} \pi_0^\intercal H_j^i \left(h_{j,j+1} e_j q_{j+1}^\intercal\right) P^{k-1-i} \\
&\overset{\text{Lem. 1}}{=} \pi_0^\intercal H_j^k Q_j + \sum_{i=0}^{k-1} \mathbf{0}_n^\intercal P^{k-1-i} = \pi_0^\intercal H_j^k Q_j = \tilde{p}_k^\intercal.
\end{aligned}$$

Proposition 1 motivates our choice of π_0 as it not only guarantees a necessary condition for exactness, $\|\mathrm{err}_0\|_1 = 0$, but extends it to the first $j-1$ steps.

Definition 2 (Initial exactness). *We call an aggregation* initially exact*, or more precisely,* $(j-1)$-exact*, if $\|\mathrm{err}_k\|_1 = 0$ for $0 \le k \le j-1$.*

Remark 1. Lemma 2 and Proposition 1 imply

$$\|\mathrm{err}_k\|_1 = \left\| \sum_{i=j-1}^{k-1} \pi_0^\intercal H_j^i \left(h_{j,j+1} e_j q_{j+1}^\intercal\right) P^{k-1-i} \right\|_1,$$

providing a closed-form formula for $\|\mathrm{err}_k\|_1$ in Arnoldi aggregations.

Proposition 2. *An Arnoldi aggregation of size j with the Krylov subspace $\mathcal{K}^j(p_0, P)$ being invariant under P is exact.*

Proof. Dynamic-exactness follows by Eq. (2) with invariance of $\mathcal{K}^j(p_0, P)$, and $p_0 = \tilde{p}_0$ follows through Proposition 1.

Theorem 1. *An Arnoldi aggregation of size j is a smallest $(j-1)$-exact aggregation of P and p_0.*

Proof. Assume we are given $P \in \mathbb{R}^{n\times n}$, $p_0 \in \mathbb{R}^n$, and an arbitrary $(j-1)$-exact aggregation with disaggregation matrix A and aggregated transient vectors π_k. Let A_{arn} be the disaggregation matrix of the corresponding Arnoldi aggregation of size j with rows q_i. By initial exactness, we must have for $0 \leq k \leq j-1$ that $\pi_k^\intercal A = p_k^\intercal$. Thus,

$$\begin{aligned} &\operatorname{span}\{\, p_0^\intercal, \ldots, p_{j-1}^\intercal \,\} \subseteq \operatorname{rowsp}(A) \\ \implies &\operatorname{span}\{\, q_1^\intercal, \ldots, q_j^\intercal \,\} = \operatorname{span}\{\, p_0^\intercal, \ldots, p_{j-1}^\intercal \,\} \subseteq \operatorname{rowsp}(A). \end{aligned}$$

As by definition $\operatorname{rowsp}(A_{\text{arn}}) = \operatorname{span}\{\, q_1^\intercal, \ldots, q_j^\intercal \,\}$, the Arnoldi aggregation of size j is indeed of minimal size because $q_1^\intercal, \ldots, q_j^\intercal$ are linearly independent.

Theorem 2. *An Arnoldi aggregation of size j with $\mathcal{K}^j(p_0, P)$ being invariant under P is a smallest exact aggregation of P and p_0.*

Proof. Follows analogous to Theorem 1. Given an exact aggregation of arbitrary size m with disaggregation matrix A, we know that $\operatorname{span}\{\, p_0^\intercal, p_1^\intercal, \ldots \,\} \subseteq \operatorname{rowsp}(A)$. As $q_1^\intercal, \ldots, q_j^\intercal$ is a basis of $\operatorname{span}\{\, p_0^\intercal, p_1^\intercal, \ldots \,\}$, the Arnoldi aggregation of size j with $q_1^\intercal, \ldots, q_j^\intercal$ as rows of the disaggregation matrix is minimal, i.e., $j \leq m$.

3.2 Determining Convergence

Recall that in Algorithm 1, we have left the criterion for termination open. Naturally, with the definition of exactness requiring $\|H_j Q_j - Q_j P\|_\infty = 0$ and Eqs. (1) and (2), $\|H_j Q_j - Q_j P\|_\infty = 0$, $|h_{j,j+1}| = 0$, or $\|r_{j+1}\|_1 = 0$ offer themselves as intuitive measurements for detecting exactness.

Unfortunately, none of these approaches work in general, which we illustrate by looking at an example DTMC. The RSVP model is a stochastic process algebra model from [23]. It is composed of a lower channel submodel with capacity M, an upper channel submodel with capacity N, and some instances of mobile nodes making call requests at a constant rate. For $M = 7$, $N = 5$ and three mobile nodes, resulting in 842 states, symmetries among these nodes enable an exact aggregation with $j = 234$ for the DTMC resulting from uniformisation of the original CTMC with uniformisation rate 30.01 [16, Sec. 6.4]. Then, by Theorem 2, Algorithm 1 must find an exact Arnoldi aggregation at size $j = 234$ or lower. Figure 1 shows the proposed naive termination criteria for $p_0 = (1, 0, \ldots, 0)^\intercal \in \mathbb{R}^{842}$, for which an exact aggregation as described exists. Note that $|h_{j,j+1}|$ is multiplied by 15 in Fig. 1 simply to match the size of the other two criteria. Since there is obviously no striking behaviour near $j = 234$, these three criteria are not suitable as termination criteria for Algorithm 1. Therefore, we must take a different approach altogether.

By [16, Thm. 4 (i)] and Proposition 1, the error bound

$$\|\mathrm{err}_k\|_1 \leq \sum_{j=0}^{k-1} \langle |\pi_k|, |H_j Q_j - Q_j P| \cdot \mathbf{1}_n \rangle \tag{3}$$

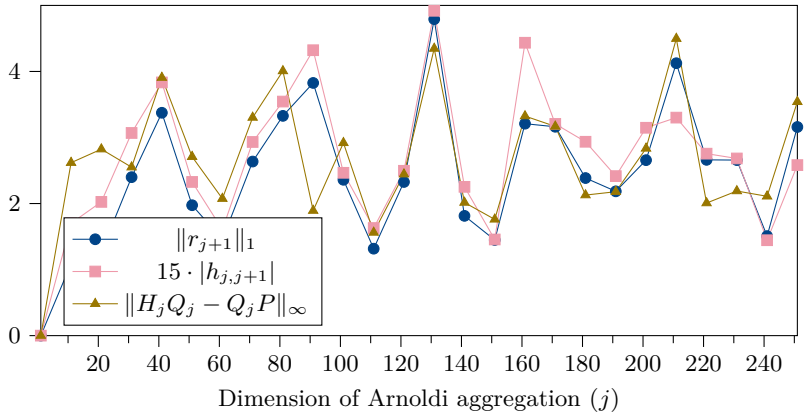

Figure 1. Different values related to $\|H_jQ_j - Q_jP\|_\infty$ depending on the dimension of Arnoldi aggregations of the RSVP model.

holds for all Arnoldi aggregations. Note that, in general, Eq. (3) cannot be improved. Although Eq. (3) offers the tightest general bound, its computational cost is too high for a usable criterion to determine whether to stop the expansion of an Arnoldi aggregation.

For an exact aggregation, it must hold that $\langle|\pi_k|, |H_jQ_j - Q_jP| \cdot \mathbf{1}_n\rangle = 0$ for all k. Under the assumption that $(\pi_k)_{k\in\mathbb{N}}$ converges to an eigenvector π, it must then hold that $\langle|\pi|, |H_jQ_j - Q_jP| \cdot \mathbf{1}_n\rangle = 0$. Importantly, this is not an equivalence and the assumption about the convergence of $(\pi_k)_{k\in\mathbb{N}}$ does not have to hold either. Still, we propose to use $\langle|\pi|, |H_jQ_j - Q_jP| \cdot \mathbf{1}_n\rangle \leq \varepsilon$ for some given $\varepsilon \in \mathbb{R}_0^+$ as a stopping criterion since it appears to work well in practice.

Per [17, Sec. 6.7], the largest eigenvalues of H_j eventually yield very good approximations of the largest eigenvalues of P for large enough j. In this case, both H_j and P, as a transition matrix, will have largest eigenvalue one. Thus, we use the Krylov-Schur method presented in [22] to determine π with eigenvalue one. If there are multiple such π, we choose the one whose eigenvalue is numerically closest to one. If π has complex entries, hinting at improper convergence, we immediately further expand the aggregation. In practice, this happens only at j so small that the resulting aggregation is unusable.

This results in Fig. 2, where we either used random p_0 or initial distributions for which we know that there is an exact aggregation at $j = 234$. Furthermore, we used ten samples per data point and $\varepsilon_{\text{mach}}$ is the IEEE 754 double precision rounding machine epsilon. As such, there is a visible correlation between $\|\text{err}_k\|_1$ and $\langle|\pi|, |H_jQ_j - Q_jP| \cdot \mathbf{1}_n\rangle$, although no formal proof or connection between these two is known. Still, over a range of models (see Section 4), we were not able to find a case where the general trend, as seen, does not hold.

The algorithm to compute Arnoldi aggregations with $p_k \approx \tilde{p}_k$ is the same as Algorithm 1 but with line 11 replaced, as seen in Algorithm 2. The convergence criterion is naively applied only every tenth expansion to reduce the runtime.

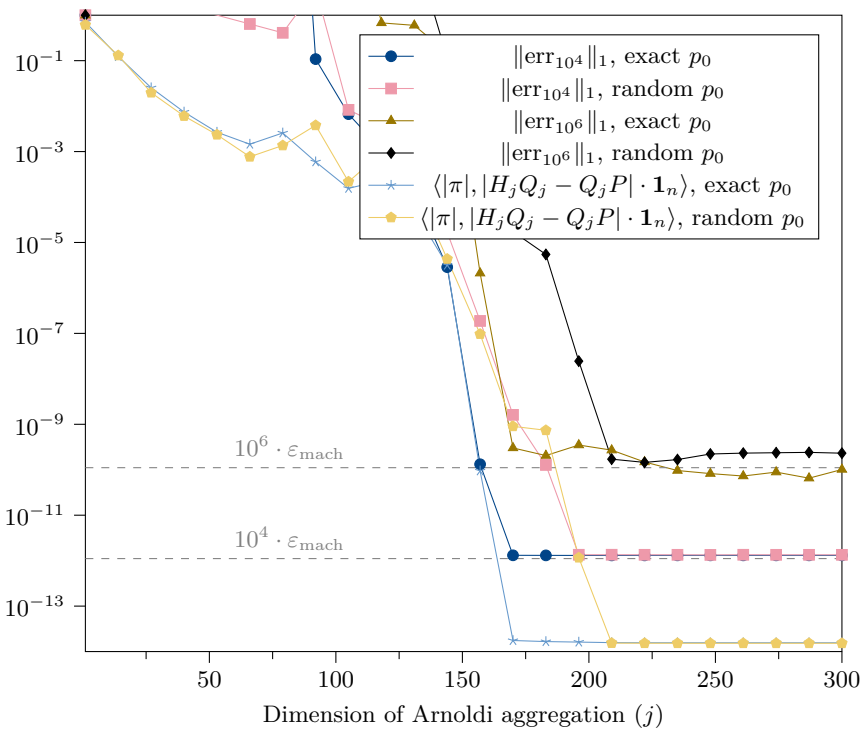

Figure 2. Proposed convergence criterion and error after 10^4 and 10^6 steps in the RSVP model, depending on the dimension of the Arnoldi aggregation.

We can also easily analyse the runtime. If P is dense, the vector-matrix multiplication in line 5 is in $\mathcal{O}(n^2)$, if P is sparse, it is in $\mathcal{O}(n)$ instead. The outer for loop is, by definition, repeated j times and the inner one on average $\frac{j}{2}$ times. Furthermore, the inner product in the inner for loop takes $\mathcal{O}(n)$. Thus, without the convergence criterion, we are in $\mathcal{O}(n^2 j + nj^2)$ for dense P or $\mathcal{O}(nj^2)$ for sparse P. In the computation needed for the convergence criterion, the calculation of π is the dominant factor. There, the Arnoldi iteration is applied to H_j a finite number of times (independent of H_j and j) with a random starting vector in our implementation. As H_j is dense, this is in $\mathcal{O}(j^2)$. Although further steps are taken to compute π, they all end up in constant time, even though with a fairly large constant. Overall, we are in $\mathcal{O}(n^2 j + nj^2 + j^3)$ for dense P and $\mathcal{O}(nj^2 + j^3)$ for sparse P. The memory complexity is the same, asymptotically. In our experience, the bottleneck is the runtime and not the memory requirement.

4 Reviewing Arnoldi Aggregations

Now that we have devised Algorithm 2, evaluating its performance is of great interest. An implementation in Julia (see [4]) version 1.11.2, using the package

Algorithm 2 Finding an Arnoldi aggregation with $\langle |\pi|, |H_j Q_j - Q_j P| \cdot \mathbf{1}_n \rangle \leq \varepsilon$

Let $\varepsilon \in \mathbb{R}_0^+$, $p_0 \in \mathbb{R}^n$, $P \in \mathbb{R}^{n \times n}$.
$q_1 \coloneqq \frac{p_0}{\|p_0\|_2}$
$Q_1 \coloneqq \begin{pmatrix} | \\ q_1^\mathsf{T} \\ | \end{pmatrix}$
for $j = 1, 2, \ldots$ **do**
    $r_1^\mathsf{T} \coloneqq q_j^\mathsf{T} P$
    ... ▷ *Algorithm 1, lines 6–10*
    if $j \pmod{10} = 0$ **then**
        Compute dominant eigenvector π^T of H_j as in [22]
        $\pi = \frac{\pi}{\|\pi^\mathsf{T} Q_j\|_1}$
        if $\pi \in \mathbb{R}^j$ and $\langle |\pi|, |H_j Q_j - Q_j P| \cdot \mathbf{1}_n \rangle \leq \varepsilon$ **then**
            $\pi_0 \coloneqq (\|p_0\|_2, 0, \ldots, 0)^\mathsf{T} \in \mathbb{R}^j$
            return H_j, Q_j, π_0, π
    ... ▷ *Algorithm 1, lines 12 and 13*

KrylovKit [10] for the underlying Arnoldi iteration and for the Krylov-Schur method from [22], can be found at [21]. Using this implementation, we compare Algorithm 2 to the Exlump algorithm [16, Algorithm 3]. This algorithm works by forming an explicit partition of the original state space, such that for any two states within the same aggregate, their incoming probabilities are close in some way. We use a so-called uniform disaggregation matrix A in Exlump aggregations, where $A(i, j)$ is the reciprocal of the size of the ith aggregate [16, p. 14].

Apart from the already introduced RSVP model, we compare these two algorithms on a range of Markov chains. For one, we use a Lotka-Volterra model (see [9] for example) with a maximum population of 100, resulting in 10,201 states, which is uniformised at a rate of 2,078. We also look at a Markov chain with 15,540 states, modelling two workstation clusters with 20 workstations per cluster, where each workstation can fail and be repaired [11]. There, the uniformisation rate is 50.08. Finally, our largest Markov chain is given by a prokaryotic gene expression model [12] with a maximal population size of five and a state space of 43,957, uniformised at a rate of 16.78. Note that the Exlump algorithm was implemented in Python, whereas we compute Arnoldi aggregations by using Julia. Thus, runtimes cannot be compared directly between these two.

We compare transient errors of the RSVP model in Fig. 3, where we see that Exlump aggregations perform similarly to Arnoldi aggregations, with the first usable aggregations at $j \approx 130$. The only exception occurs with random p_0 after 10^4 steps in Exlump aggregations. This is due to Exlump aggregations being formed independently of p_0, but with only specific p_0 enabling exact aggregations.

In contrast to the similar error performance, the runtime analysis in Fig. 4 shows significant differences. Note that the different line heights indicating the time needed to compute p_{10^5} without aggregation are caused by Exlump and Algorithms 1 and 2 being implemented in different programming languages. In Fig. 4 and the following figures, Algorithm 1 is run for some pre-specified number of iterations without any termination criterion. Exlump allows to compute $\tilde{p}_{10^5}$

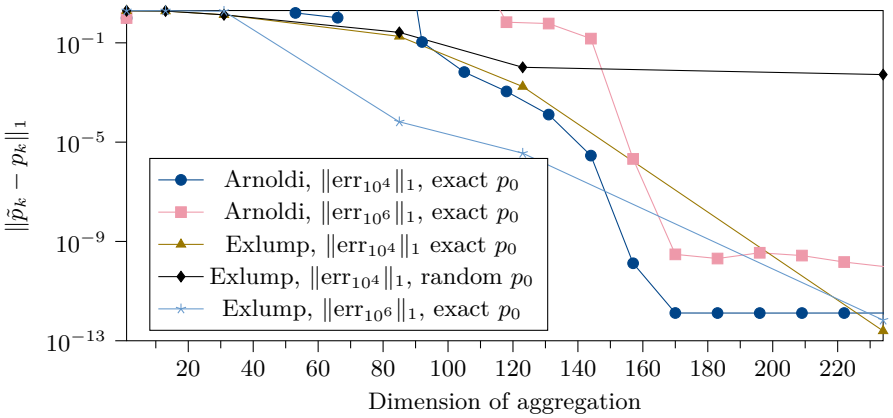

Figure 3. $\|\text{err}_{10^4}\|_1$ and $\|\text{err}_{10^6}\|_1$ in the RSVP model for Arnoldi and Exlump aggregations, with different initial distributions, depending on the aggregation dimension.

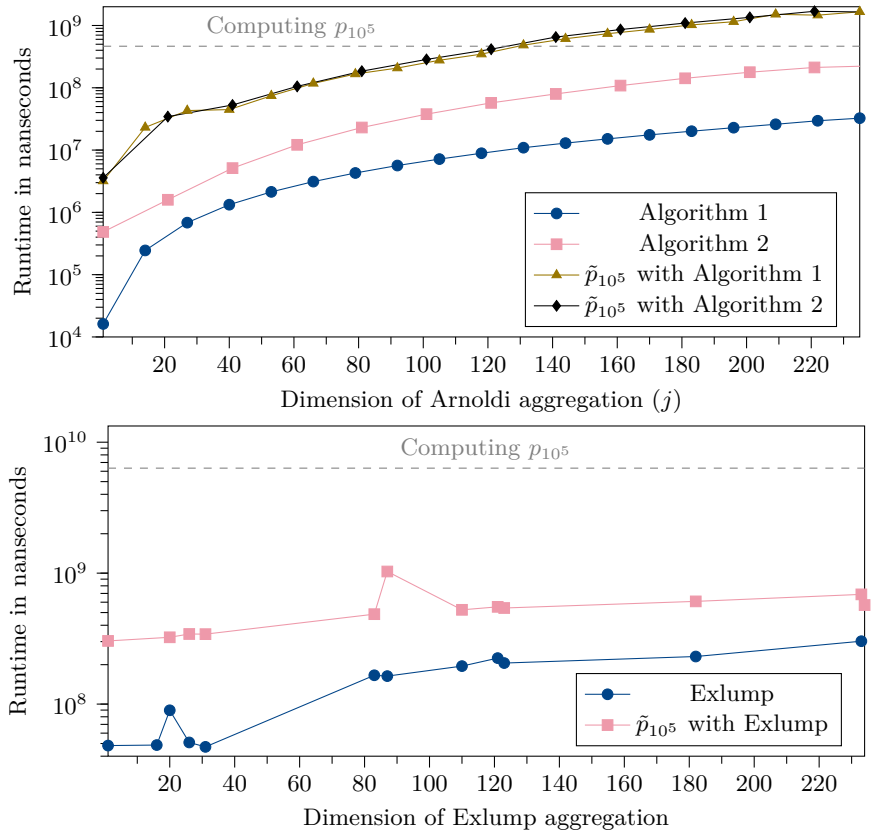

Figure 4. Runtime of computing Arnoldi and Exlump aggregations of the RSVP model, both with and without computing $\tilde{p}_{10^5}$, depending on the dimension of the aggregations.

consistently faster than computing p_{10^5} naively. While the Exlump algorithm itself is, relatively speaking, faster and scales better for larger aggregation sizes than Algorithms 1 and 2, it also offers a more important advantage: the Π of Exlump aggregations retains the sparsity of P, whereas the H_j of an Arnoldi aggregation is dense by definition. Thus, computing approximated transient distributions is much faster for Exlump aggregations. In comparison, it accounts for around 85% of the total runtime needed to compute $\tilde{p}_{10^5}$ in Arnoldi aggregations. The criterion from Algorithm 2 takes around 12% of the total runtime with the remaining 3% needed for the actual underlying Arnoldi iteration.

We continue with the errors in the workstation cluster model in Fig. 5, where the behaviour is different. Arnoldi aggregations have already reached

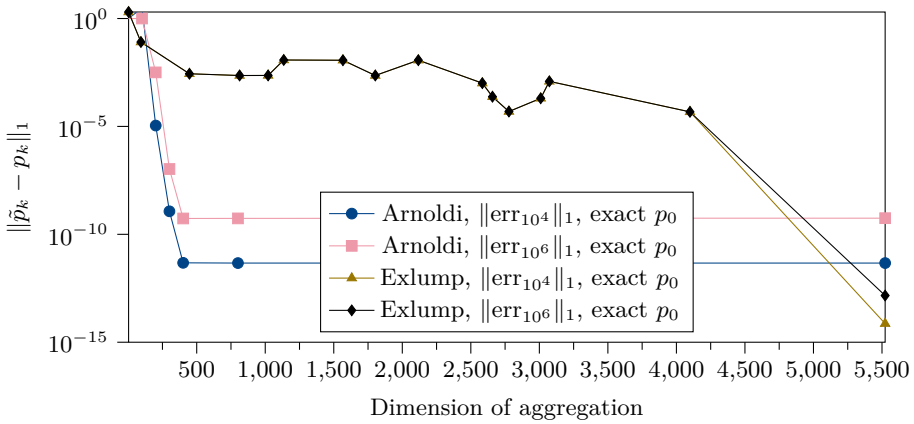

Figure 5. $\|\mathrm{err}_{10^4}\|_1$ and $\|\mathrm{err}_{10^6}\|_1$ in the workstation cluster model for Arnoldi and Exlump aggregations, depending on the aggregation dimension.

their optimum at $j \approx 400$, beating Exlump aggregations by far until their exact aggregation at size 5,523. Notably, in Arnoldi aggregations we consistently have $k\varepsilon_{\mathrm{mach}} \approx \|\mathrm{err}_k\|_1$ for $j \gtrsim 400$, while Exlump aggregations can eventually achieve a lower error. This can be explained by $\tilde{p}_k$ converging toward the actual stationary distribution in Exlump aggregations (because the actual stationary distribution is compatible with the exact aggregation with 5,523 states, see [6, Theorem 3]), which is not guaranteed for Arnoldi aggregations of the size considered here. Generally speaking, random p_0 yield the same effect as shown here for those p_0 which enable an exact aggregation of dimension 5,523.

While computing $\tilde{p}_k$ still takes the most time for the workstation cluster model in Fig. 6 with the Arnoldi aggregation, its share went down to about 45% with the convergence criterion now needing roughly 40% of the total runtime. Still, the results are much better than in the RSVP model. Exlump aggregations fail to consistently beat the naive approach for computing transient distributions,

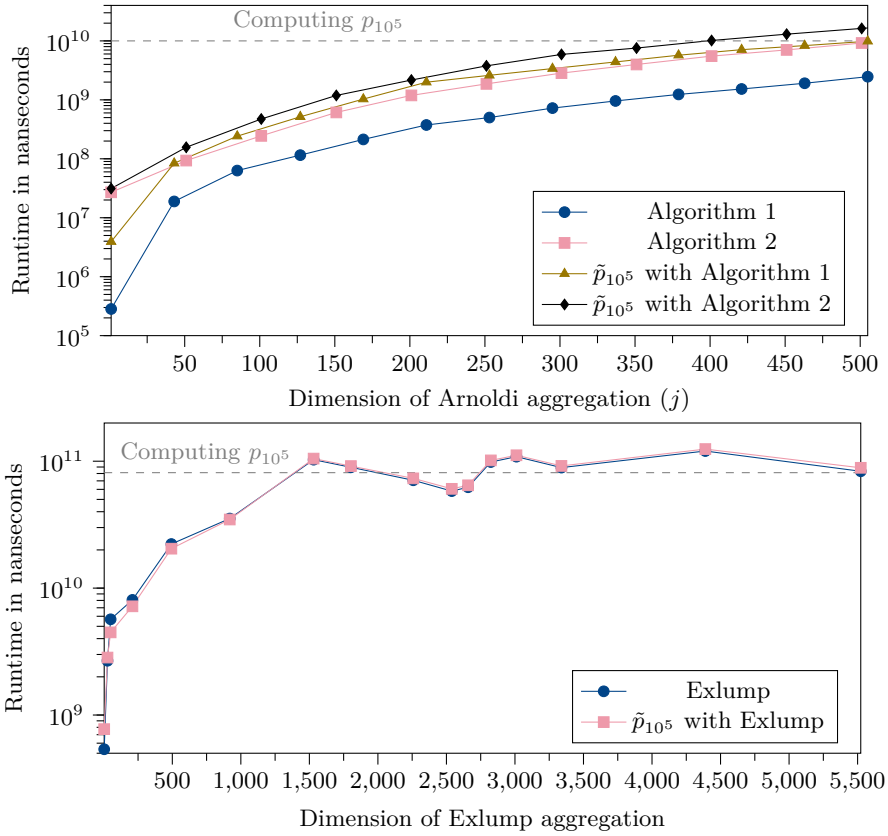

Figure 6. Runtime of computing Arnoldi and Exlump aggregations of the workstation cluster model, both with and without computing $\tilde{p}_{10^5}$, depending on the dimension of the aggregations.

whereas Algorithm 2 is faster up to $j \approx 400$. An Arnoldi aggregation with $j \approx 300$ has $\|\text{err}_{10^5}\|_1 \approx 10^{-8}$, while being almost four times faster than computing p_{10^5}.

Errors for our largest model, the gene expression model, can be seen in Fig. 7. The only usable Exlump aggregation is the one with $j = n$, thus disqualifying Exlump for this model, while Arnoldi aggregations quickly improve and reach their optimum at around $j \approx 1{,}100$, again limited by rounding errors.

The same holds if we look at the runtime needed to reduce the state space of the gene expression model, as in Fig. 8. We did not plot the runtime needed by Exlump aggregations, as they failed to produce any acceptable errors, as seen in Fig. 7. With such a large model, the dense vector-matrix multiplication forced by a dense H_j in Arnoldi aggregations is no longer dominant. Instead, the convergence criterion takes up most of the runtime, around 75%. $j \approx 450$ is the last dimension of the Arnoldi aggregation where we see a speed-up. However, an error of $\|\text{err}_{10^5}\|_1 \approx 10^{-1}$ at this size makes it irrelevant for practical use.

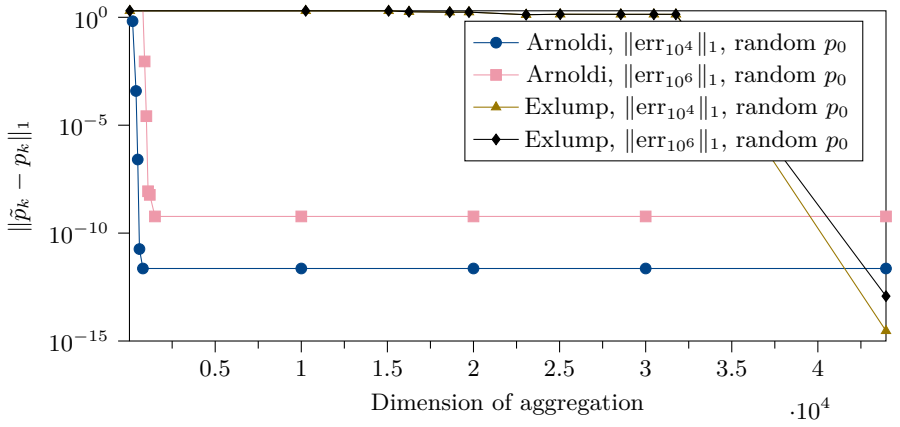

Figure 7. $\|\mathrm{err}_{10^4}\|_1$ and $\|\mathrm{err}_{10^6}\|_1$ in the gene expression model for Arnoldi and Exlump aggregations, depending on the aggregation dimension.

Lastly, although not plotted, we get to the Lotka-Volterra model. It mirrors the behaviour of the gene expression model at a smaller size, as Exlump again fails to find a usable aggregation except for the trivial aggregation of size n, while Arnoldi aggregations converge very soon, although still too late to be fast enough for practical applications.

5 Conclusion

Results Definition 1 defines an Arnoldi iteration based aggregation of a DTMC that incorporates the initial distribution p_0. Theorems 1 and 2 then show that, under exact arithmetic, any (initially) exact Arnoldi aggregation is of minimal size. In practice, however, numerical instability in the Arnoldi iteration (see Section 3.2) prevents exactness. Instead, we adopt the convergence criterion $\langle|\pi|, |H_j Q_j - Q_j P| \cdot \mathbf{1}_n\rangle \leq \varepsilon$, although choosing ε remains heuristic, since no direct bound links $\|\mathrm{err}_k\|_1$ to $\langle|\pi|, |H_j Q_j - Q_j P| \cdot \mathbf{1}_n\rangle$. Empirically, the relationship in Fig. 2 holds across all tested models.

In experiments, small Arnoldi aggregations suffer from dense H_j, such that computing $\tilde{p}_k$ has cost $\mathcal{O}(kj^2)$, whereas computing p_k costs only $\mathcal{O}(kn)$ for sparse P. As the model size grows, computing the eigenvector π of H_j eventually dominates the cost of the dense vector-matrix product. Overall, Algorithm 2 runs in $\mathcal{O}(n^2 j + nj^2 + j^3)$ for dense P and in $\mathcal{O}(nj^2 + j^3)$ for sparse ones. Despite these costs, we observed substantial speed-ups at low error bounds for the workstation cluster model, outperforming Exlump aggregations in this case. Moreover, Arnoldi aggregations consistently produce non-trivial aggregations, whereas Exlump sometimes fails to do so.

Outlook We see several avenues for future work. One is to evaluate our method against additional aggregation techniques (e.g., [1,5,7,16]) over a broader suite of

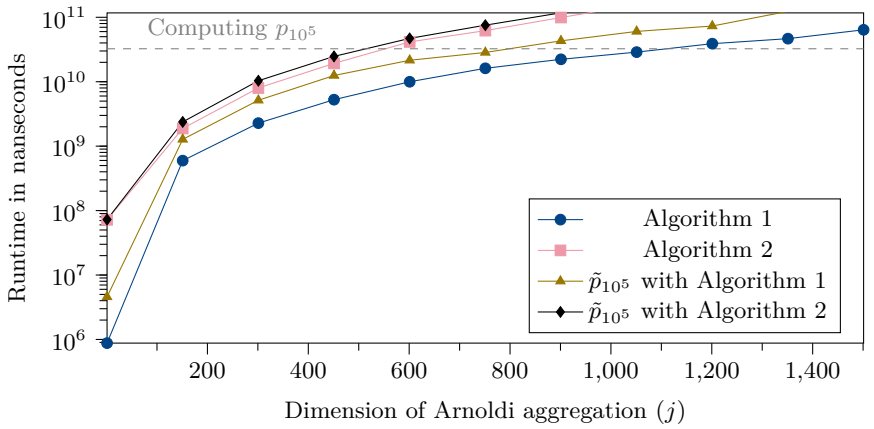

Figure 8. Runtime of computing Arnoldi aggregations of the gene expression model, both with and without computing $\tilde{p}_{10^5}$, depending on the dimension of the aggregation.

models and a wider range of time points. A second is to speed up Algorithm 2 by dynamically tuning how often the convergence criterion $\langle |\pi|, |H_j Q_j - Q_j P| \cdot \mathbf{1}_n \rangle \leq \varepsilon$ is checked. Expanding on this, when the criterion is first satisfied at dimension $j + \ell$ but not at j, one can exploit the recursive structure of $H_{j+\ell}$ and $Q_{j+\ell}$ along with the Krylov-Schur approach of [22] to identify an intermediate aggregation dimension j' (with $j < j' \leq j + \ell$) that still meets the criterion at lower computational cost for $\tilde{p}_k$. Lastly, the Arnoldi iteration itself could be optimized via parallel or randomized Gram-Schmidt variants (e.g., [3,15]), improved memory management, or integration with high-performance libraries such as ARPACK [14].

References

1. Abate, A., Andriushchenko, R., Češka, M., Kwiatkowska, M.: Adaptive formal approximations of Markov chains. Performance Evaluation **148**, 102,207 (2021), https://doi.org/10.1016/j.peva.2021.102207
2. Arnoldi, W.E.: The principle of minimized iterations in the solution of the matrix eigenvalue problem. Quarterly of Applied Mathematics **9**(1), 17–29 (1951), https://doi.org/10.1090/qam/42792
3. Balabanov, O., Grigori, L.: Randomized Gram-Schmidt process with application to GMRES. SIAM Journal on Scientific Computing **44**(3), A1,450–A1,474 (2022), https://doi.org/10.1137/20M138870X
4. Bezanson, J., Edelman, A., Karpinski, S., Shah, V.B.: Julia: A fresh approach to numerical computing. SIAM review **59**(1), 65–98 (2017), https://doi.org/10.1137/141000671
5. Bittracher, A., Schütte, C.: A probabilistic algorithm for aggregating vastly undersampled large Markov chains. Physica D: Nonlinear Phenomena **416**, 132,799 (2021), https://doi.org/10.1016/j.physd.2020.132799

6. Buchholz, P.: Exact and ordinary lumpability in finite Markov chains. Journal of Applied Probability **31**(1), 59–75 (1994), https://doi.org/10.2307/3215235
7. Buchholz, P., Kriege, J.: Approximate aggregation of Markovian models using alternating least squares. Performance Evaluation **73**, 73–90 (2014), https://doi.org/10.1016/j.peva.2013.09.001, special Issue on the 9th International Conference on Quantitative Evaluation of Systems
8. Courtois, P.J.: Decomposability: Queueing and Computer System Applications. Academic Press, ACM monograph series (1 1977), https://doi.org/10.1016/C2013-0-07302-1
9. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry **81**(25), 2,340–2,361 (12 1977), https://doi.org/10.1021/j100540a008
10. Haegeman, J.: KrylovKit (version 0.9.4) (2025), https://doi.org/10.5281/zenodo.14832110
11. Haverkort, B., Hermanns, H., Katoen, J.P.: On the use of model checking techniques for dependability evaluation. In: Proceedings 19th IEEE Symposium on Reliable Distributed Systems SRDS-2000. pp. 228–237 (2000), https://doi.org/10.1109/RELDI.2000.885410
12. Kierzek, A.M., Zaim, J., Zielenkiewicz, P.: The effect of transcription and translation initiation frequencies on the stochastic fluctuations in prokaryotic gene expression. Journal of Biological Chemistry **276**(11), 8,165–8,171 (7 2001), https://doi.org/10.1074/jbc.M006264200
13. Koury, J.R., McAllister, D.F., Stewart, W.J.: Iterative methods for computing stationary distributions of nearly completely decomposable Markov chains. SIAM Journal on Algebraic Discrete Methods **5**(2), 164–186 (1984), https://doi.org/10.1137/0605019
14. Lehoucq, R., Sorensen, D.C., Yang, C.: ARPACK Users' Guide. Society for Industrial and Applied Mathematics (4 1998), https://doi.org/10.1137/1.9780898719628
15. Lingen, F.J.: Efficient Gram-Schmidt orthonormalisation on parallel computers. Communications in Numerical Methods in Engineering **16**(1), 57–66 (2000), https://doi.org/10.1002/(SICI)1099-0887(200001)16:1<57::AID-CNM320>3.0.CO;2-I
16. Michel, F., Siegle, M.: Formal error bounds for the state space reduction of Markov chains. Performance Evaluation **167**, 102,464 (2025), https://doi.org/10.1016/j.peva.2024.102464
17. Saad, Y.: Krylov Subspace Methods, chap. 6, pp. 125–162. Society for Industrial and Applied Mathematics, 2 edn. (2011), https://doi.org/10.1137/1.9781611970739.ch6
18. de Souza e Silva, E., Gail, H.R.: Transient Solutions for Markov Chains, pp. 43–79. Springer US, Boston, MA (2000), https://doi.org/10.1007/978-1-4757-4828-4_3
19. Simon, H.A., Ando, A.: Aggregation of variables in dynamic systems. Econometrica **29**(2), 111–138 (1961), https://doi.org/10.2307/1909285
20. Sonnentag, P.: Finding the smallest possible exact aggregation of a Markov chain (2025), https://doi.org/10.48550/arXiv.2507.11157
21. Sonnentag, P.: MarkovChainAggregations (version 0.1.0) (2025), https://doi.org/10.5281/zenodo.16082617
22. Stewart, G.W.: A Krylov-Schur algorithm for large eigenproblems. SIAM Journal on Matrix Analysis and Applications **23**(3), 601–614 (2002), https://doi.org/10.1137/S0895479800371529
23. Wang, H., Laurenson, D.I., Hillston, J.: Evaluation of RSVP and mobility-aware RSVP using performance evaluation process algebra. In: 2008 IEEE International Conference on Communications. pp. 192–197 (2008), https://doi.org/10.1109/ICC.2008.43