Sergey V. Dolgov and Eugene E. Tyrtyshnikov

On evolution of solution times for the chemical master equation of the enzymatic futile cycle

Abstract: We investigate three tensor product numerical data compression techniques in solution of the chemical master equation for the enzymatic futile cycle and compare them with the previously reported results, obtained by the stochastic simulation algorithm. On this particular example from systems biology, we show the history how the newly proposed tensor product methods reduced the computational complexity of the futile cycle modelling from days on a HPC cluster to hours and even minutes on a workstation.

Keywords: High-dimensional problems, alternating linear scheme, density matrix renormalization group, chemical master equation.

MSC 2010: 15A69, 33F05, 65F10

Sergey V. Dolgov: Institute of Numerical Mathematics, Russian Academy of Sciences, Moscow 119333, Russia. Max Planck Institute for Dynamics of Complex Technical Systems,39106 Magdeburg, Germany. E-mail: sergey.v.dolgov@gmail.com.
Eugene E. Tyrtyshnikov: Institute of Numerical Mathematics, Russian Academy of Sciences, Moscow 119333, Russia. E-mail: eugene.tyrtyshnikov@gmail.com.

1 Introduction

The chemical master equation (CME) is an important probabilistic model in systems biology [7, 8, 23]. When cell processes are considered, typical numbers of molecules (copy numbers) of reacting substances (such as RNAs, proteins, etc.) can be as few as tens. In such situations, stochastic fluctuations in the cell behaviour, resulting from random occurrence of reactions, become significant [2, 22]. The CME describes the joint probability density that each copy number takes a particular value at a particular moment of time.

Let d active substances in a well-mixed medium be given. We denote their copy numbers as integer indices $i_1, \dots, i_d$, where each $i_k \geq 0$, $k = 1, \dots, d$. Generally, arbitrarily large values are possible, but very large copy numbers have usually very small probability. This allows to truncate the index values to finite ranges, $i_k = 0, \dots, n_k - 1$ (see [16]).

We assume that substances react via M reactions. Each reaction is characterized by the stoichiometric vector $\mathbf{z}^m \in \mathbb{Z}^d$, showing the difference between numbers of molecules produced and consumed by the reaction, and the reaction rate function $w^m(i_1, \dots, i_d) \in \mathbb{R}_+$, showing how fast the reaction goes in time, provided that the copy numbers equal $i_1, \dots, i_d$. For brevity, we will agglomerate several indices into a multiindex, $\mathbf{i} = (i_1, \dots, i_d)$. The joint probability distribution is denoted as $\psi(\mathbf{i}, t)$, and the chemical master equation is written as the following linear ODE:

$$\frac{\mathrm{d}\psi(\mathbf{i}, t)}{\mathrm{d}t} = \sum_{m=1}^{M} w^m(\mathbf{i} - \mathbf{z}^m)\psi(\mathbf{i} - \mathbf{z}^m, t) - w^m(\mathbf{i})\psi(\mathbf{i}, t). \tag{1.1}$$

The difficulty of this problem is the curse of dimensionality: if all $n_k \lesssim n$, the total amount of values defining ψ scales as $\mathcal{O}(n^d)$ and rapidly goes beyond any memory limits with growing number of substances d. To estimate statistics (such as the average copy numbers $\langle i_k \rangle$) without constructing the probability function explicitly, a Monte-Carlo-type Stochastic Simulation Algorithm (SSA) was proposed [7]. It tracks many realizations of the system, and in each of them it draws the time step and the particular reaction, according to random numbers, and conducts the chosen reaction. This process is repeated until the desired modelling time is reached, and the desired quantity is averaged over realizations. According to the big numbers law, the number of realizations required for the accuracy ε grows as ε^{-2}, which can be less than n^d, but still too large.

An alternative way is to compress the probability function values in some low-parametric representation. This work started from the sparse grids approach [9], followed by tensor product algorithms [1, 3, 10, 14]. This paper is also devoted to the latter class.

Tensor product algorithms are based on separation of variables on the discrete level. One of the simplest, but robust representations is the Matrix Product States [19, 21], also known as Tensor Train (TT) format [18],

$$\psi(i_1,\dots,i_d) = \sum_{\alpha_1=1}^{r_1}\cdots\sum_{\alpha_{d-1}=1}^{r_{d-1}} \psi^{(1)}_{\alpha_1}(i_1)\psi^{(2)}_{\alpha_1,\alpha_2}(i_2)\cdots\psi^{(d-1)}_{\alpha_{d-2},\alpha_{d-1}}(i_{d-1})\psi^{(d)}_{\alpha_{d-1}}(i_d). \tag{1.2}$$

We see that each element of a multiindex array ψ is represented (or approximated) by a sum of products of elements of two- and three-index arrays $\psi^{(k)}$, called TT factors. The auxiliary summation indices $\alpha_1,\dots,\alpha_{d-1}$ are called rank indices, and their ranges $r_1,\dots,r_{d-1}$ are called TT ranks. The TT ranks depend on particular ψ and accuracy, if (1.2) is satisfied approximately. If it is possible to limit $r_k \lesssim r$, $k = 1,\dots,d-1$, the TT factors consume $\mathcal{O}(dnr^2)$ memory, which can be much less than n^d. Usually we refer to r as the maximal TT rank.

The TT format allows to compress a given ψ up to the tolerance ε, also if ψ is already given as a TT format, but possibly with overestimated TT ranks. This happens if we sum two arrays in the TT format, or perform the matrix multiplication. In the CME, we consider $\psi(t)$ as a vector with elements $\psi(\mathbf{i},t)$, and the right-hand side of (1.1) can be written as a matrix-vector product $A\psi(t)$, meaning

$$\sum_{\mathbf{j}} A(\mathbf{i},\mathbf{j})\psi(\mathbf{j},t) = \sum_{j_1=0}^{n_1-1}\cdots\sum_{j_d=0}^{n_d-1} A(i_1,\dots,i_d;\, j_1,\dots,j_d)\psi(j_1,\dots,j_d,t).$$

The matrix A can be also represented in the following TT format,

$$A(\mathbf{i},\mathbf{j}) = \sum_{\beta_1=1}^{R_1}\cdots\sum_{\beta_{d-1}=1}^{R_{d-1}} A^{(1)}_{\beta_1}(i_1,j_1)A^{(2)}_{\beta_1,\beta_2}(i_2,j_2)\cdots A^{(d)}_{\beta_{d-1}}(i_d,j_d)$$

which is consistent with the Kronecker product $\otimes$ if $d = 2$ and $R_1 = 1$.

Implicit time schemes require us to solve large linear systems on ψ. A state of the art approach to this problem is alternating tensor product algorithms [11, 21, 24]. Given the system $B\psi = g$, we subsequently fix all TT factors of ψ and update only one or two. This renders $B\psi = g$ an overdetermined system on elements of the sought factors, which is resolved with the fixed factors. The Density Matrix Renormalization Group (DMRG) methods from quantum physics [12, 24] and newly developed Alternating Minimal Energy (AMEn) algorithm [6] are efficient. The DMRG approach seeks two neighbouring TT factors in each step, for example, k and $k+1$, which allows to adapt the TT rank r_k to the desired accuracy. While DMRG method was found to be extremely effective for spin Schrödinger eigenvalue problem, it may return a wrong solution of linear systems, especially with non-symmetric matrices. The AMEn method is usually faster and more robust, since it seeks only one TT factor in each step, but then performs an explicit augmentation of the computed factor by the TT factor of the current residual. This allows to change TT ranks and facilitate convergence. Both DMRG and AMEn algorithms were developed in Matlab in the framework of the TT-Toolbox [17].

We investigate both methods in solution of the chemical master equation for the enzymatic futile cycle model, which we describe in Section 2. We also compare two different implicit time schemes: an hp-Discontinuous Galerkin discretization, employed in the previous work [14] and a simple Euler scheme with many time steps aggregated into a single large linear system, as proposed in [4]. Details on these schemes are given in Section 3. In Section 4 we present numerical experiments and demonstrate that the AMEn method, applied to the Euler scheme, delivers solution of the same quality ten times faster than the previous *hp*-DG scheme with the DMRG method, since the former allows much larger time steps. Finally, comparing our results with the previously reported SSA calculations, which required 1500 cores on a distributed HPC system, we may conclude that the new tensor product methods reduce the time consumption of the CME modelling from days to minutes, making it really practical.

2 Model

We consider the enzymatic futile cycle model [20]. It contains 6 species: E^f_+, E^b_+, X, $X^\star$, E^b_- and E^f_-, whose copy numbers are denoted as $i_1, \dots, i_6$, respectively. The reaction network is as follows:

$$\begin{aligned} X + E^f_+ &\rightleftarrows E^b_+, & X^\star + E^f_0 &\rightleftarrows E^b_- \\ E^b_+ &\rightarrow E^f_+ + X^\star, & E^b_- &\rightarrow E^f_- + X \end{aligned} \tag{2.1}$$

with the stoichiometry vectors and the reactions rates:

$$\begin{aligned} w^1(\mathbf{i}) &= 40 \cdot i_1 i_3, & \mathbf{z}^1 &= \begin{bmatrix} -1 & 1 & -1 & 0 & 0 & 0 \end{bmatrix} \\ w^2(\mathbf{i}) &= 10^4 \cdot i_2, & \mathbf{z}^2 &= \begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 \end{bmatrix} \\ w^3(\mathbf{i}) &= 10^4 \cdot i_2, & \mathbf{z}^3 &= \begin{bmatrix} 1 & -1 & 0 & 1 & 0 & 0 \end{bmatrix} \\ w^4(\mathbf{i}) &= 200 \cdot i_4 i_6, & \mathbf{z}^4 &= \begin{bmatrix} 0 & 0 & 0 & -1 & 1 & -1 \end{bmatrix} \\ w^5(\mathbf{i}) &= 100 \cdot i_5, & \mathbf{z}^5 &= \begin{bmatrix} 0 & 0 & 0 & 1 & -1 & 1 \end{bmatrix} \\ w^6(\mathbf{i}) &= 5000 \cdot i_5, & \mathbf{z}^6 &= \begin{bmatrix} 0 & 0 & 1 & 0 & -1 & 1 \end{bmatrix}. \end{aligned}$$

The index ranges of indices $\mathbf{i}$ are restricted as follows,

$$i_1, i_2, i_5, i_6 = 0, \dots, 3, \qquad i_3, i_4 = 0, \dots, 127 \tag{2.2}$$

which corresponds to the total CME size 2^{22}.

The initial state of the CME is chosen as a direct product of unitary vectors,

$$\psi_0 = \delta_3 \otimes \delta_1 \otimes \delta_{31} \otimes \delta_{91} \otimes \delta_1 \otimes \delta_3 \tag{2.3}$$

which means that the initial copy numbers equal $2, 0, 30, 90, 0, 2$ with probability 1.

The CME is modelled for the time interval $T = 1$.

3 Numerical schemes

We compare three tensor product solution schemes. In all cases, we use implicit discretizations in time, posed for several time layers simultaneously. That is, if $\psi(t)$ denotes the vector of 2^{22} distribution values at the given t, we will denote by ψ a stack of $\psi(t)$ for several time steps (or polynomial coefficients). The whole time interval $[0, 1]$ is split into subintervals $[t_{p-1}, t_p]$, $p = 1, \dots, P$, where $t_0 = 0$, $t_P = 1$, and intermediate steps will be specified below. Therefore, for each interval $[t_{p-1}, t_p]$ we are to solve linear systems of the form

$$B\psi = g, \qquad B = I \otimes G - A \otimes M, \quad g = \psi(t_{p-1}) \otimes s \tag{3.1}$$

where A is the CME matrix, I is the corresponding identity matrix of size 2^{22}, G is the stiffness matrix, M is the mass matrix, and s is the right-hand side, related to time.

The first scheme is the hp-DG-QTT discretization [14, 15], combined with the DMRG method for the linear system solution [5, procedure dmrg_solve3 in TT-Toolbox]. Matrices G, M and s correspond to the Legendre basis of degree 3 and have the following forms:

$$G = \begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \end{bmatrix}, \qquad M = \frac{t_p - t_{p-1}}{2} \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2/3 & 0 & 0 \\ 0 & 0 & 2/5 & 0 \\ 0 & 0 & 0 & 2/7 \end{bmatrix}, \qquad s = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix}.$$

The time splitting is done as follows. We set $h = 5 \times 10^{-4}$, $T_1 = 0.3$. Then on $[0, h]$ we introduce 10 geometrically graded steps $t_p = (2^p - 1) \cdot 1023^{-1} h$, $p = 0, \dots, 10$. On $(h, T_1]$ we use equidistant mesh $t_p =$

$h(p{-}9)$, $p = 11, \dots, 9{+}T_1/h$. Finally, on $(T_1, T]$ we increase the mesh geometrically, $t_p = T_1 \left(1 - h/T_1\right)^{9+T_1/h-p}$, $p = 10 + T_1/h, \dots, P$, which results in $P = 1331$ steps total with $t_P \approx 1.0004$.

Given the vector of coefficients ψ of (3.1) in the Legendre basis, we may enumerate it as a matrix $\psi(\mathbf{i}, j)$, where $j = 1, \dots, 4$ corresponds to the time variable, and then the last snapshot is recovered as $\psi(\mathbf{i}, t_p) = \sum_{j=1}^{4} \psi(\mathbf{i}, j)$.

The second approach is the implicit Euler scheme, solved by the same DMRG algorithm. In each time interval we introduce 2^L snapshots, and the matrices in (3.1) are the following: $G = t_p - t_{p-1} \cdot 2^{-L} \cdot \operatorname{tridiag}(-1, 1, 0) \in \mathbb{R}^{2^L \times 2^L}$, M is the identity matrix of size 2^L, and s is the first unitary vector. The last snapshot $\psi(t_p)$ is recovered trivially as the last 2^{22} coefficients in ψ.

Since the QTT format allows us to take large L, we also consider larger time intervals. On $[0, 0.02]$ we introduce $t_p = 10^{-3}\, p$, $p = 0, \dots, 20$, then on $(0.02, 0.05]$ we increase the step size, $t_p = 0.02{+}5{\times}10^{-3}(p{-}20)$ for $p = 21, \dots, 26$, on $(0.05, 0.1]$ we use $t_p = 0.05 + 10^{-2}(p - 26)$, $p = 27, \dots, 31$, and finally, on $(0.1, 1]$ the time grid has the form $t_p = 0.1 + 0.1(p - 31)$, $p = 32, \dots, 40$. In all time intervals we choose $L = 14$, which corresponds to the effective time steps ranging from 6×10^{-8} in the beginning of the process to 6×10^{-6} in the last interval $[0.1, 1]$, and gives sufficient discretization accuracy. The reason for this experiment is to show that the DMRG can be applied for large time intervals.

The third approach uses the same Euler time discretization, but the AMEn algorithm [6, procedure amen_solve2 in TT-Toolbox] is employed instead of the DMRG to solve (3.1).

In order to reconstruct the experiment from [14] exactly, we used the dmrg_solve3 routine rolled back to the state of July 12, 2012. Some details were taken from the preprint version [13]. Among them there are the time splitting and tuning parameters of the DMRG: the GMRES method in local DMRG steps was conducted with the Krylov basis size 50 and 2 restarts, the random enrichment rank was set to 2, and the maximal number of DMRG sweeps is 5. The same parameters are used in the Euler scheme. In the AMEn algorithm, the local GMRES solver was also limited to 50 inner and 2 outer iterations, the number of sweeps was not limited, but the enrichment rank, which is more important in the AMEn method, was set to 4. The amen_solve2 routine is up to November 10, 2014. In all experiments, the solution from the previous time interval was passed as an initial guess to the DMRG and AMEn methods, and the residual stopping threshold was set to $\varepsilon = 10^{-6}$.

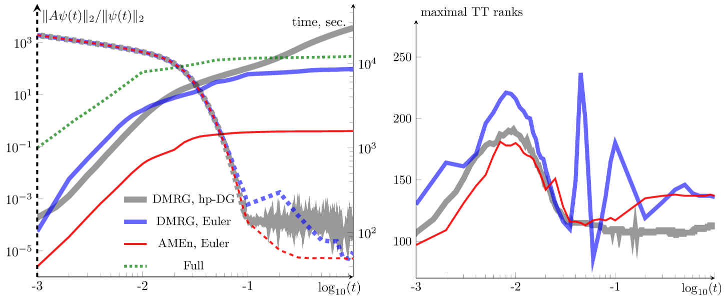
In the fourth experiment, we solve the CME with the full storage of the solution and sparse matrix, since the size 2^{22} fits into the memory. However, the system (3.1) on many time snapshots would still be too large, so we integrate the CME via the standard Crank-Nicolson scheme with the following time steps: 2×10^{-6} on $[0, 0.01]$, 2×10^{-5} on $(0.01, 0.05]$, 2×10^{-4} on $(0.05, 0.1]$, and 2×10^{-3} on $(0.1, 1]$. The linear system in the implicit stage is solved by the BiCGStab method with the residual threshold 10^{-13}.

4 Results

In Fig. 1 we show cumulative computational times, norms of the residual (see Fig. 1a) and maximal ranks in the QTT representations of the solution (see Fig. 1b) versus the system time. Four methods 'DMRG, hp-DG', 'DMRG, Euler', 'AMEn, Euler', and 'Full' are according to the description above. The QTT format is applied for lexicographic order of indices $i_1, \dots, i_6$ and their binary digits, which corresponds to the experiment (D) in [14]. Correctness of the problem setting can be verified by the initial residual, $\|A\psi_0\|_2 / \|\psi_0\|_2 \approx 5.27{\times}10^4$, the decay of the residual with time, and the evolution of the TT ranks, which coincide with the previous results.

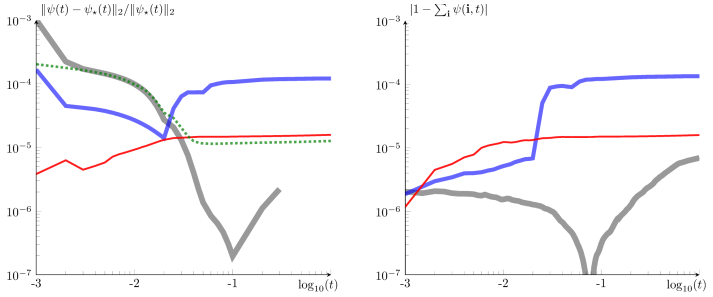
Experiments were conducted in a Linux Matlab R2012a on a single core of Intel Xeon E5504 CPU at 2 GHz. The total CPU time of the 'DMRG, hp-DG' approach is 2.61×10^4 seconds. Comparing this with 1.52×10^4 in [14], we estimate the rescaling ratio 1.7146 due to different CPUs and versions of Matlab and other libraries. This ratio is used to normalize and compare CPU times in Table 1.

Focusing on experiments with the Euler scheme, we observe that the DMRG is also applicable here, but in the transient region it suffers from large condition numbers of the local problems. This is reflected by jumps in the TT ranks (Fig. 1b), since inaccurate solution of local problems introduces some noise. Nevertheless, larger time steps allow to decrease the total CPU time.



(a) (b)

Figure 1. (a) cumulative CPU times (sec.) and residual $\|A\psi(t)\|/\|\psi(t)\|$; (b) maximal TT ranks.



(a) (b)

Figure 2. (a) errors w.r.t. the reference solution; (b) probability deficiency.

Table 1. CPU times, normalized to the results in [14].

SSA	DMRG, *hp*-DG	Full	DMRG, Euler	AMEn, Euler
2×10^8	1.52×10^4	7.05×10^3	4.97×10^3	9.23×10^2

More robust is the AMEn approach, which is more than 10 times faster than the *hp*-DG method. Smaller and better conditioned local problems allow the AMEn method to avoid the outlined problem of noisy solution and deliver a more accurate result.

Interestingly, the full scheme is not the slowest one here: sparse matrix operations are quite efficient, which makes them faster than the DMRG method, if the TT ranks are large.

Other indicators of the solution correctness are the difference w.r.t. the reference solution (Fig. 2a) and the degeneracy of the probability normalization (Fig. 2b). The reference solution $\psi_\star$ is computed by the 'AMEn, Euler' approach with the residual tolerance 10^{-8}. We may notice that fine time intervals in the *hp*-DG scheme yield pretty accurate results in terms of both indicators, but this is overweighted by a large cost. The AMEn, Euler technique gives a comparable accuracy on average, while being significantly faster.

The full scheme behaves similarly to the *hp*-DG test: it exhibits a larger error in the beginning of the process due to large time steps, but converges to a more accurate solution in the end.

Funding: The work was supported by the Russian Science Foundation (Project 14-11-00806).

References

[1] A. Ammar, E. Cueto, and F. Chinesta, Reduction of the chemical master equation for gene regulatory networks using proper generalized decompositions. *Int. J. Numer. Meth. Biomed. Engrg.* **28** (2012), No. 9, 960–973.

[2] A. Arkin, J. Ross, and H. H. McAdams, Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected Escherichia coli cells. *Genetics* **149** (1998), No. 4, 1633–1648.

[3] S. Dolgov and B. Khoromskij, Simultaneous state-time approximation of the chemical master equation using tensor product formats. *Numer. Linear Algebra Appl.* (2014) (to appear).

[4] S. V. Dolgov, B. N. Khoromskij, and I. V. Oseledets, Fast solution of multi-dimensional parabolic problems in the tensor train/quantized tensor train–format with initial application to the Fokker–Planck equation. *SIAM J. Sci. Comput.* **34** (2012), No. 6, A3016–A3038.

[5] S. V. Dolgov and I. V. Oseledets, Solution of linear systems and matrix inversion in the TT-format. *SIAM J. Sci. Comput.* **34** (2012), No. 5, A2718–A2739.

[6] S. V. Dolgov and D. V. Savostyanov, Alternating minimal energy methods for linear systems in higher dimensions. *SIAM J. Sci. Comput.* **36** (2014), No. 5, A2248–A2271.

[7] D. T. Gillespie, A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys.* **22** (1976), No. 4, 403–434.

[8] D. T. Gillespie, A rigorous derivation of the chemical master equation. *Physica A: Stat. Mech. Appl.* **188** (1992), No. 1–3, 404–425.

[9] M. Hegland, C. Burden, L. Santoso, S. MacNamara, and H. Booth, A solver for the stochastic master equation applied to gene regulatory networks. *J. Comput. Appl. Math.* **205** (2007), No. 2, 708–724.

[10] M. Hegland and J. Garcke, On the numerical solution of the chemical master equation with sums of rank one tensors. *ANZIAM* **52** (2011), C628–C643.

[11] S. Holtz, T. Rohwedder, and R. Schneider, The alternating linear scheme for tensor optimization in the tensor train format. *SIAM J. Sci. Comput.* **34** (2012), No. 2, A683–A713.

[12] E. Jeckelmann, Dynamical density–matrix renormalization-group method. *Phys. Rev. B* **66** (2002), 045114.

[13] V. Kazeev, M. Khammash, M. Nip, and C. Schwab, Direct solution of the chemical master equation using quantized tensor trains. *Research Report No. 04*. SAM, ETH Zürich, 2013.

[14] V. Kazeev, M. Khammash, M. Nip, and C. Schwab, Direct solution of the chemical master equation using quantized tensor trains.. *PLOS Comput. Biology* **10** (2014), No. 3, e100359.

[15] V Kazeev, O Reichmann, and Ch. Schwab, *hp*-DG-QTT solution of high-dimensional degenerate diffusion equations. *Tech. Report No. 2012-11*. ETH SAM, Zürich, 2012.

[16] B. Munsky and M. Khammash, The finite state projection algorithm for the solution of the chemical master equation. *J. Chem. Phys.* **124** (2006), 044104.

[17] I. Oseledets, S. Dolgov, D. Savostyanov, and V. Kazeev, TT-Toolbox v2.2, 2014. URL http://github.com/oseledets/TT-Toolbox.

[18] I. V. Oseledets, Tensor-train decomposition. *SIAM J. Sci. Comput.* **33** (2011), No. 5, 2295–2317.

[19] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac, Matrix product state representations. *Quantum Info. Comput.* **7** (2007), No. 5, 401–430.

[20] M. Samoilov, S. Plyasunov, and A. P. Arkin, Stochastic amplification and signaling in enzymatic futile cycles through noise-induced bistability with oscillations. *Proc. Nat. Acad. Sci.* **102** (2005), No. 7, 2310–2315.

[21] U. Schollwöck, The density–matrix renormalization group. *Rev. Mod. Phys.* **77** (2005), No. 1, 259–315.

[22] R. Steuer, Effects of stochasticity in models of the cell cycle: from quantized cycle times to noise-induced oscillations. *J. Theor. Biology* **228** (2004), No. 3, 293–301.

[23] N. G. van Kampen, *Stochastic Processes in Physics and Chemistry*. North Holland, Amsterdam, 1981.

[24] S. R. White, Density–matrix algorithms for quantum renormalization groups. *Phys. Rev. B* **48** (1993), No. 14, 10345–10356.

Received October 20, 2014; accepted October 28, 2014.