PARALLEL COMPUTING FOR MARKOV CHAINS WITH ISLANDS AND PORTS

by

Amod Jung Basnet

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Applied Mathematics

Charlotte

2017

Approved by:

Dr. Isaac M. Sonin

Dr. Michael Grabchak

Dr. Jaya Bishwal

Dr. Arun Ravindran

ABSTRACT

AMOD JUNG BASNET. Parallel computing for Markov chains with islands and ports. (Under the direction of DR. ISAAC M. SONIN)

We present recursive algorithms to calculate invariant distributions and fundamental matrices of Markov chains specified by the "Islands & Ports" (IP) model. The state space of the IP model can be partitioned into "islands" and "ports". An island is a group of states with potentially many connections inside of the island but a relatively small number of connections between islands. The states connecting different islands are called "ports". Our algorithm is developed in the framework of the "state reduction approach" but the special structure of the state space allows computations to be performed in *parallel*.

ACKNOWLEDGMENTS

First and foremost, I am grateful to Dr. Isaac M. Sonin without whose guidance this thesis would not have been possible. The algorithms presented in this thesis are only possible because of his expertise in the field, his willingness to let me use his published works, his insightful approach to problem solving, and his joy for discovery. I am incredibly humbled to collaborate with him and contribute in every way possible. This thesis is an embodiment of his passion for academia and his dedication to mentorship, even during difficult times.

I would like to thank my committee members Dr. Jaya Bishwal, Dr. Michael Grabchak, and Dr. Arun Ravindran for their remarks and helpful discussions. In particular, I am thankful to my committee members for their feedback during our presentation at the probability seminar. I also wish to thank Dr. Wei Cai, and his research group, for warmly receiving us at their weekly seminar.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1 Motivation

Discrete Time Markov Chains (DTMCs), or simply Markov Chains (MCs), are used to model various systems of interest. Recently there has been a growing interest in the study of MCs with "large" and "very large" state spaces. These MCs appear in applications such as Web search, genetic modeling, and queueing theory. The state spaces of such MCs may have thousands, and even millions of states. For example, in population genetics, the so-called Wright-Fisher model involves transition matrices that can be of size $(2N+1) \times (2N+1)$, $N \in [20, 1000]$ (see e.g. [22]). In Web search, Google's PageRank algorithm uses a $n \times n$ transition matrix, P, to rank Web pages. The transition matrix is given by $P = cM + (1-c)U$, where $U = \frac{1}{n}ee^T$ is a uniform matrix, e is a vector of all ones, $c \in (0, 1)$, and M is the hyperlink matrix (see e.g. [17, 2]). According to Google, $c = 0.85$, which is the probability that a surfer jumps to a random page. The matrix, P, is an example of a very large ergodic MC with billions of Web pages as nodes. The PageRank vector, π, is the invariant distribution of the MC specified by the matrix P. The entries of the vector, π, approximates the long run probability that a random surfer visits a particular Web page. In other words, it ranks the Web pages based on the surfer's random behavior. A survey of various other modifications like updating and accelerating the calculation of the PageRank

vector can be found, for example, in [17]. Google uses the power method to compute π and claims that the method converges after about 50 iterations. Because the MC described by P is ergodic, power method ensures that P converges to a limiting matrix, $A = e\pi^T$, almost surely.

Although we can use classical methods, like the power method, to analyze large MCs, such methods may be inefficient for large state spaces. As an alternative, we can develop methods that exploit specific structures that may be present in the state spaces. For example, we can partition the state spaces of some MCs into "clusters". These are groups of states such that states inside each group have strong connections, but each group is only weakly connected to other groups. In such cases, we can use cluster-based methods to calculate various characteristics of these MCs. In the cluster-based approach, we calculate the "local" characteristics of each cluster first, and then use them to approximate the"global" characteristics for the whole chain (see e.g. [19]). In the literature, MCs with such structure are said to be "nearly uncoupled"[1] and the cluster-based approach is also called the aggregation/disaggregation (AD) approach. Here, we refer to a Markov model that specifies nearly uncoupled MCs as the "nearly uncoupled model".

Following this approach, numerous algorithms have been proposed to approximate the invariant distribution, π (see e.g. [6, 7]). A similar approach is also taken in the so-called the BlockRank algorithm — a modified version of the PageRank algorithm to compute the PageRank vector — developed after experiments indicated clustering of Web pages in the form of "hosts" (see e.g. [14]). A host page is a Web page

[1]These MCs are also known as Nearly Completely Decomposable (NCD) MCs.

inside of which a large number of other Web pages sit. Such pages include university domains like www.uncc.edu, the IBM domain, and various news sites. Variants of the AD approach to compute the PageRank vector, for example, the partial aggregation method (PAM), fast two-stage algorithm (FTSA) and others, are summarized in [2]. In addition, the effect of analytic perturbation on the PageRank vector can also be found, for example, in [2, 1, 3].

To apply cluster-based methods, clusters must first be identified in the state spaces. There are many methods to identify clusters for MCs (see e.g. [18, 20]). However, they are not always accurate and only give an approximate number of clusters, yet these cluster-based methods provide a convenient way to analyze a large MC. Moreover, calculations can be done even faster if we can take advantage of parallel computing. By "parallel computing" we mean using multiple processors to perform calculations involving, in our case, a large transition matrix.

2 The Islands and Ports (IP) Model

In this thesis, we study finite, ergodic Markov models with large "state spaces". Let $M = (S, P)$ be a Markov model. Here, S is the state space and P is the transition matrix. The state space is finite and discrete, $S = \{1, 2, \ldots, n\}, n < \infty$, and it can be partitioned into k disjoint *"islands"*, L_i, so that $S = \bigcup_{i=1}^{k} L_i$. Each island, L_i, is also a disjoint union of *"interior"* states, R_i, and *"ports"*, T_i, i.e. $L_i = R_i \bigcup T_i$. We allow transitions between two states to occur only if these states belong to the same island or to ports, i.e. if $p(i, j) > 0$, then either $i, j \in L_r$ or $i \in T_s, j \in T_r$, for some $1 \leq r, s \leq k$. Let T be the union of all ports, $T = \bigcup_{j=1}^{k} T_j$ and $|T| = t$. We assume

that t is of the same order as the size of the biggest island and, for simplicity, we assume that $t = |L_i| = m, 1 \leq i \leq k$. Figure 1 below is an example for the case when $k = 3$.

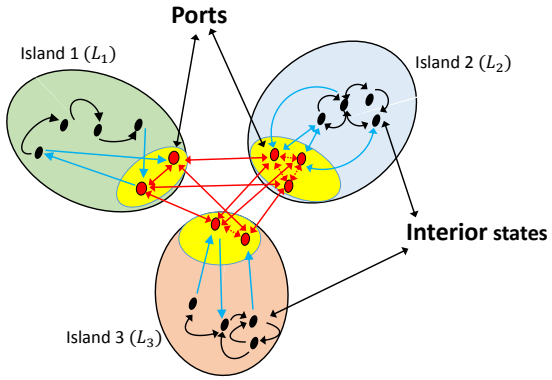


Figure 1: An example of an IP model with 3 islands.

The configuration of states in our model as "islands" and "ports" is a special type of clustering that can be found in the state spaces of many Markov models describing many real scenarios. For example, if we consider trading zones as islands and large ports — limited in number— as ports, then we can use our model to analyze trade routes of modern day tankers or large container ships. We can also apply our model to study interactions between international and multinational agencies. Such agencies have many local branches (in countries, regions, etc.) but interactions across agencies only occur through the central governing bodies. In fact, our model can be used to describe any multi-layered, diversified organization (e.g. financial networks such as banks) because communication channels in such organizations tend to be structured in a similar way. In order to emphasize this unique configuration of states in the state space of our model, we refer to such a model as the "Islands & Ports" (IP) model.

The parallel feature of this algorithm also allows making external changes inside of an island without significantly affecting the calculations in other islands. For example, we can adjust transition probabilities of states inside each island or change the number of states in each island. Parallel features of the algorithm are also preserved even if we introduce new islands into our model. In this sense, updating our algorithm to new information is relatively easy.

Our model is different from the nearly decoupled model that we mentioned before. In our model, transitions between two islands can occur only through the ports. Unlike the transitions in nearly decoupled MCs, transitions between any two islands in our models are certainly more restrictive but these transitions do not have to be weak. This means that the islands L_i and L_j, $i \neq j$, in our model can be strongly or weakly linked to each other. In this sense, our model is quite general. If all the transitions between islands are close to zero, then the IP model reduces to a nearly uncoupled model.

Our main objective is to develop algorithms to calculate essential characteristics for MCs specified by the IP model. The characteristics we are interested in are given by the invariant distribution, π, and the fundamental matrix, N. Both of these characteristics are important in applications. We already know that the distribution, π, gives the long run probability distribution of a MC. The entries of the matrix, N, gives the expected number of visits to states inside a certain (nonabsorbing) subset of the state space before exiting it at some finite time. By taking advantage of the assumptions we made about the state space in our model, our algorithms allow most of the calculations involved in computing these characteristics to be performed in

parallel.

3 Layout of the Dissertation

The outline of the dissertation is as follows: In Chapter 2, we present some preliminary results. In particular, we discuss the so-called State Reduction (SR) approach. The SR approach is the building block in all our algorithms. In Chapter 3, we discuss the IP model and develop a three-stage recursive algorithm to calculate π for this model. We refer to our algorithm as the *IP algorithm.*

(Algorithm 1: The IP algorithm) This is a recursive algorithm that calculates π in three stages. Even though it is a sequential algorithm, calculations for the most expensive stages — stages 1 and 3 of the algorithm can be done in parallel. This algorithm has time complexity, as per our argument, of order $(\frac{n}{k})^3$, where k is the total number of islands and n is the total number of states in the model. In comparison, solving for π using the GTH/S algorithm, which is the most accurate (direct) algorithm, has time complexity of order n^3.

There exists a rich literature on relatively new algorithms to compute π for nearly uncoupled MCs (see e.g. [16, 33, 6]). To our knowledge, none of these algorithms allow parallel computing to the extent our IP algorithm does. In addition, most of these methods are iterative whereas our algorithm gives exact results.

Because transition probabilities in our model are more restrictive than that in the nearly uncoupled models, we also discuss the possibility of extending our results to approximate invariant distributions for the nearly uncoupled MCs. For this, we perform some preliminary simulations with our model. We also discuss the effect

of our approximations using previously known bounds. These bounds appear in the perturbation theory of stochastic matrices. To elaborate this further, the invariant distributions of MCs under perturbations have been rigorously studied since the early 1960s. These perturbations include linear perturbations (see e.g. [24, 11, 5]) as well as analytical perturbations (see e.g. [3]). Under linear perturbations, most bounds for π include the so-called fundamental matrix, V, for ergodic MCs. A comparison of the bounds for linear perturbations can also be found, for example, in [5]. Our approximations of the nearly uncoupled model only includes a simple case of linear perturbation.

There are many algorithms to calculate the matrix V (see e.g. [13, 30, 10]). However, the fundamental matrix, N, for transient MCs are not studied as extensively. One of the reason why this is so may be due to the fact that calculation of N is simpler. The calculation of N involves a matrix inversion. If the state space is large, inversion is computationally expensive. There are few recursive algorithms proposed to calculate such matrices; (see e.g. [12]).

In Chapter 4, we develop an algorithm to compute N for any substochastic matrix Q. We name this algorithm the FUNDQ algorithm.

(Algorithm 2: The FUNDQ algorithm) This recursive algorithm consists of two stages: In the forward stage, we apply the SR approach to eliminate a subset of states from the matrix Q. The backward stage involves computing a sequence of fundamental matrices using the recursive formula given in Proposition 4.6. The time complexity of this algorithm is approximately of order $k(n^2) + (n-k)^3, 1 \leq k < n$, where n is the number of states in the subset being considered.

We also apply the FUNDQ algorithm to develop another algorithm to calculate the fundamental matrix N defined for a specific substochastic matrix Q in the IP model.

(Algorithm 3: The IP FUND algorithm) This is a recursive algorithm which has a similar three stage structure as that in the IP algorithm. The calculations in stages 1 and 3 of the algorithm can also be done in parallel.

Additionally, we also discuss another set of fundamental matrices, $N_{(1)}, N_{(2)}, \ldots, N_{(k)}$, where k is the number of islands in the IP model, that result, without any extra calculations, from the IP algorithm in Chapter 3. In the Appendix, we give some proofs to the (previously published) lemmas and propositions we use in this thesis. We also provide pseudo-codes to the three algorithms that we present in Chapters 3 and 4 in the Appendix.

CHAPTER 2: THE STATE REDUCTION APPROACH

Let $M = (S, P)$ be a homogeneous, ergodic Markov model, where $S = \{1, 2, \ldots, n\}$, $n \leq \infty$, is the discrete state space, and P is the transition matrix. In this chapter, we discuss the state reduction (SR) approach. This approach was first used in the GTH/S algorithm (see section 6) to compute π for model M. Since then, it has been used to recursively calculate many important characteristics of MCs. It served as the building block in many algorithms such as those used to calculate fundamental matrices of transient MCs (see e.g. [12]) and for ergodic MCs (see e.g. [30]), in matrix-analytic methods for block-structured matrices (see e.g. [34, 8]), to solve optimal stopping problems (see e.g. [26, 31, 32]; [28, 29]), and to calculate mean passage times (see e.g. [13]). The SR approach also serves as the building block in our algorithms in Chapters 3 and 4. Our discussion of the SR approach in this chapter is similar to that given in [27] and [29]. The SR approach has a two-stage structure:

1. (**Forward stage**) Let MC (Z_n) be specified by model M. And let $D \subseteq S$. Then we can obtain another MC (Y_n) (see Lemma 2.1) from the model M by eliminating set D from S. The path of the MC (Y_n) coincides with that of the original MC (Z_n) in remaining set $S \setminus D$. The MC (Y_n) is called a *censored* MC specified by the D-reduced model $M_D = (S \setminus D, P_D)$, where the matrix P_D is

the new transition matrix for model M_D.

The concept of censored MCs is an insightful idea of Kolmogorov and Doëblin. According to this idea, the censored chain observed in the subset of the original state space is still a MC. The states that do not belong to the observation set of the censored MC are said to be *eliminated*. One can calculate various characteristics for MC (Y_n).

2. (**Backward stage**) States that were previously eliminated from set D in the forward stage are *"inserted"* or *"restored"*. Using results calculated for the censored MC (Y_n), one can also calculate the characteristics of the original MC (Z_n).

4 State Reduction (SR)

Let MC (Z_n) be a S-valued MC specified by model M with some initial distribution μ_0. If we let $D \subseteq S$, $C = S \setminus D$, we can decompose matrix P as the first matrix below

$$P = \begin{array}{c} \begin{matrix} \scriptstyle (D) & \scriptstyle (C) \end{matrix} \\ \begin{bmatrix} Q & T \\ R & K \end{bmatrix} \end{array}, \quad P^D = \begin{array}{c} \begin{matrix} \scriptstyle (D) & \scriptstyle (C) \end{matrix} \\ \begin{bmatrix} 0 & NT \\ 0 & P_D \end{bmatrix} \end{array}, \quad W^D = \begin{array}{c} \begin{matrix} \scriptstyle (D) & \scriptstyle (C) \end{matrix} \\ \begin{bmatrix} QN & NT \\ RN & P_D \end{bmatrix} \end{array}, \tag{2.1}$$

where the substochastic matrices $Q := \{p(x,y) : x, y \in D\}, T := \{p(x,y) : x \in D, y \in C\}, R := \{p(x,y) : x \in C, y \in D\}$, and $K := \{p(x,y) : x, y \in C\}$ describe transitions inside of set D, from sets D to C, from sets C to D, and inside of set C, respectively. Let $\tau_0 = 0$ and $\tau_{n+1} = \min\{n > \tau_n : \ Z_n \in C\}$. Then $\tau_0, \tau_1, \tau_2, \ldots$ are the times of the zero, the first, the second, and so on, *visits* to set C. Let $Z_0 \in C$ and consider

the random sequence ($Y_n = Z_{\tau_n}$). According to Lemma 2.1, (Y_n) is a censored MC specified by model $M_D = (C, P_D)$. The transition matrix P_D is calculated using formula (2.2).

Lemma 2.1 follows from the strong Markov property and standard probabilistic reasoning. We credit this Lemma to the works of Kolmogorov and Doëblin (see e.g. [27]).

Lemma 2.1. *(SR Lemma)*

The random sequence (Y_n) *is a Markov chain in model* $M_D = (C, P_D)$*, where the transition matrix* $P_D = \{p(x, y) : x, y \in C\}$ *is given by the formula*

$$P_D = K + RU = K + RNT. \tag{2.2}$$

Remark 2.2. The matrix $U = N_D T$ (see Lemma 4.2 Chapter 4) is a matrix of the distribution of the MC (Z_n) at the time of the first exit τ_1 to C, and $N_D \equiv N$ is the fundamental matrix of the sub-stochastic matrix Q. The matrix P_D is also known as the stochastic complement of K in P (see e.g. [19]).

A proof based on probability theory is given, for example, in [15]. An algebraic proof of Lemma 2.1 can be found, for example, in [19].

We can represent the matrix P_D in formula (2.2) as

$$p_D(x, y) = p(x, y) + \sum_{j \in D} p(x, j) P_j(Z_{\tau_1} = y), \tag{2.3}$$

where $\tau_1 = \min\{n > 0 : Z_n \in C\}$. In formula (2.2), the matrix N can be calculated as $N = \sum_{n=0}^{\infty} Q^n = (I - Q)^{-1}$, where I is a $|Q| \times |Q|$ identity matrix. In Chapter 4,

we discuss the calculation of matrix N in greater detail.

Instead of calculating the matrix P_D by eliminating D in one step, thus requiring a matrix inversion to calculate N, we can, instead, eliminate it in $|D|$ steps by eliminating one state $z \in D$ at a time. To elaborate this further, let us assume that set D only consists of one non-absorbing point z. In this case, each column of the matrix $P_D = P_{\{z\}}$ in (2.2) can be written as

$$p_{\{z\}}(x, \cdot) = p(x, \cdot) + p(x, z)n(z)p(z, \cdot), \qquad x \in C, \tag{2.4}$$

where $n(z) = 1/(1 - p(z, z))$ is the one-dimensional fundamental matrix. Here, $n(z)$ gives the expected number of visits to state z by MC (Z_n) before exiting this state at time τ_1 to C. We say that the matrix $P_{\{z\}}$ is obtained from matrix P in one *iteration*. If $|D| = d$, then we can eliminate one state from D in each iteration using (2.4) and obtain P_D in d iterations. The matrix P_D describes the behavior of the censored MC (Y_n) with values only in set C.

For calculations, it is more convenient to have the initial and the reduced stochastic matrices of equal full size. For this, the matrix P_D is extended in [28, 29] to the full size $|S| \times |S|$ stochastic matrix P^D (second matrix in (2.1)) by assuming that the MC (Y_n) can have initial points in D. But after one step, it will jump to C and will remain in C. Equivalently, if $Z_0 \in D$, then the MC (Z_n) will exit at time τ_1, and for all $n > \tau_1$, it will remain in C as D is eliminated. Thus, $p^D(z, z) = 0$, for all $z \in D$. But for $z \in D$ and $y \in C$, using the first equality in formula (2.2), we obtain

$$p^D(z, y) = p(z, y) + \sum_{z' \in D} p(z, z')u(z', y). \tag{2.5}$$

By noting that $U = NT$, in matrix form, the right hand side of (2.5) equals $T+QU = T + QNT = (I + QN)T = NT$, where the last equality follows from equation (4.4) Chapter 4. For $x, y \in C$, the corresponding distribution is given by the submatrix P_D.

To obtain the matrix P^D in d iterations, we apply formula (2.4) to the full size matrices. Thus, if we denote the initial matrix P as P_1 and the matrix obtained after eliminating one state $\{z\}$ as P_2, then the columns of matrix P_2 can be written as

$$p_2(\cdot, z) = 0, \quad p_2(\cdot, y) = p_1(\cdot, y) + p_1(\cdot, z)\frac{p_1(z, y)}{1 - p_1(z, z)}, \quad y \neq z. \tag{2.6}$$

To distinguish between the matrices in models M_D and M^D, we introduce the following notation: *matrices* in reduced models are denoted by subscripts, i.e., M_D, P_D, N_D and so on; whereas *matrices* in full size model are denoted with superscripts, i.e. M^D, P^D, N^D and so on.

For computations, we prefer to calculate another full (nonstochastic) matrix W^D (third matrix in (2.1)). Before we explain why we do this, let us first explain how the matrix W^D is calculated and how it compares with previously discussed matrices P_D and P^D.

$$w_2(\cdot, y) = p_1(\cdot, y) + p_1(\cdot, z)\frac{p_1(z, y)}{1 - p_1(z, z)}, \quad y \neq z. \tag{2.7}$$

The matrix W^D is calculated by using the *elimination formula* given in (2.7) by applying formula (2.6) to all states, including all the states that were previously eliminated. According to (2.7) if we let $W_1 = P$, then we obtain matrix W_2 after eliminating one state z from the initial matrix W_1. Note that the column correspond-

ing to z is now non-zero.

If we let $D = \{1, 2, \ldots, k, \ldots, d\} \subseteq S$. The matrix W^D is obtained after eliminating set D in d iterations. In each iteration, we apply formula (2.7) to the previous matrix starting with $W_1 = P$. Let matrix W_k be the full matrix obtained after eliminating first k states from set D, then using formula (2.7), the columns of matrix W_{k+1} can be calculated as

$$w_{k+1}(\cdot, y) = w_k(\cdot, y) + w_k(\cdot, z)\frac{w_k(z, y)}{1 - w_k(z, z)}, \quad y \neq z. \tag{2.8}$$

Although the form of the matrix W^D given in (2.1) is quite transparent, proving it is quite difficult. Formula (2.7) and the interpretation of the nonzero columns in W^D is given in [28, 29]. By the definition of the matrices W^D and P^D, their columns for $y \notin D$ coincide but the matrix P^D has zero columns for $y \in D$.

From our discussion thus far, we can eliminate D and obtain the stochastic matrix, P_D, for the censored MC directly using (2.2). We can also calculate matrix P^D and obtain $P_D \subset P^D$ corresponding to states in $S \setminus D$. Instead, we prefer to calculate matrix W^D and obtain $P_D \subset W^D$. Note that $P_D \subset P^D \subset W^D$.

$$P_1 \Longrightarrow P_2 \Longrightarrow \cdots P_d \Longrightarrow P_{d+1} = P_D \ (stochastic), \tag{2.9}$$

$$P'_1 \Longrightarrow P'_2 \Longrightarrow \cdots P'_d \Longrightarrow P'_{d+1} = P^D \ (stochastic), \tag{2.10}$$

$$W_1 = P_1 \Longrightarrow W_2 \Longrightarrow \cdots W_d \Longrightarrow W_{d+1} = W^D \ (nonstochastic), \tag{2.11}$$

where the $(d+1)^{th}$ matrix is the matrix obtained after D is completely eliminated, in d iterations, from the state space S. The matrices P' and W are full matrices calculated using (2.6) and (2.7), respectively.

Remark 2.3. Unless otherwise stated, all of our calculations involve full matrices W given in (2.11). That is, the stochastic matrix for the non-eliminated states can be obtained as a submatrix of these W matrices. We obtain the matrix P_D from W^D which is the submatrix in W^D corresponding to states in $S \setminus D$.

We now explain why we prefer to use full matrices W in our calculations. In the two-stage structure of the SR approach, the forward stage involves the elimination of a set of states, say D, from S. As explained before, the matrix P_D is a stochastic matrix describing transitions in the reduced model M_D. The idea of the forward stage of the SR approach is straightforward — it allows for dimension reduction, that is, we reduce the matrix P into a smaller stochastic matrix P_D. We perform calculations using this matrix P_D. Sometimes, it is the characteristics of the original MC (Z_n) that we are interested in. This requires *inserting* states that were eliminated in the forward stage. In particular, we want to *restore* any of the eliminated states, *in any order*, without taking into account the *order* in which that state was eliminated during the forward stage. This can be achieved, starting with the matrix W^D and using the "insertion" formula (2.12) given by Lemma 2.4.

Lemma 2.4. *[28, 29] If the matrix W_{k+1} is obtained from matrix W_k by the elimination formula* (2.8)*, then the matrix W_k can be obtained from matrix W_{k+1} by*

$$w_k(\cdot, y) = w_{k+1}(\cdot, y) - w_{k+1}(\cdot, z)\frac{w_{k+1}(z, y)}{1 + w_{k+1}(z, z)}, \quad y \in S. \tag{2.12}$$

To distinguish between the matrices obtained in the forward stage by applying formula (2.8) and the matrices calculated in the backward stage by applying formula (2.12),

we denote the latter as matrix $\hat{W}$. As a consequence of the insertion formula (2.12), we obtain

$$\hat{W}_1 = W_1 = P_1 \Longleftarrow \hat{W}_2 \Longleftarrow \cdots \Longleftarrow \hat{W}_{d+1} = W^D, \tag{2.13}$$

where the full matrix $\hat{W}$, other than W^D and W_1, in (2.11) and (2.13) need not be the same because the order of elimination and insertion can vary.

The following Lemma follows from Lemma 2.1. It was also proved in [15].

Lemma 2.5. *Let $\pi = (\pi_1\ \pi_2\ ...\pi_s...\pi_n)^T$ be the invariant distribution of the original MC (Z_n). Let $|C| = s$ and let $\bar{\pi}_C = (\bar{\pi}_1\ \bar{\pi}_2\ ..\ \bar{\pi}_s)^T$ be the invariant distribution for MC (Y_n). Then*

$$\bar{\pi}_i = \frac{\pi_i}{\sum_{i\in C}\pi_i} \quad \textit{and} \quad \bar{\pi}_C P_D = \bar{\pi}_C, \quad i = 1, 2, \ldots, s. \tag{2.14}$$

In the next section, we do a straightforward analysis of the time complexity for the calculation of matrices P_D in (2.9), W_D in (2.11), and $\hat{W}$ in (2.13).

5 Order Count

Let us suppose P be a $n \times n$ matrix.

(Calculation of matrix P_D) Each step of the state elimination for the sequence given in (2.9) requires $(n-1)^2$ additions, 1 subtraction, (n^2-1) multiplications, and 1 division, so eliminating n states requires $n^3 - 2n^2 + 2n$ additions and subtractions, n^3 divisions, and multiplications. If we count only multiplication and divisions, the time complexity is of order n^3.

(Calculation of matrix W^D) Each step of the state reduction from the sequence

given in (2.11) requires n^2 additions, 1 subtraction, n^2 multiplications, and 1 division. Thus, for n steps, the time complexity is also of order n^3.

(Calculation of matrix $\hat{W}_1$) Each step of the state insertion for the sequence in (2.13) requires 1 addition, n^2 subtractions, n^2 multiplications, and 1 division. The insertion of n states requires n^3+n additions and subtractions, n^3+n multiplications and divisions. If we only count the multiplications and divisions, then the complexity is also of order n^3 .

6 Grassman, Taskar and Heyman/Sheskin (GTH/S) Algorithm

We now discuss the GTH/S proposed, independently, by Grassman, Taskar and Heyman [9], and by Sheskin in 1985 [25], to calculate π for a $n \times n$ transition matrix P of an ergodic model. The distribution $\pi^T = [\pi_1 \ \pi_2 \ \ldots \ \pi_n]$ is the solution of the linear system

$$\pi^T = \pi^T P, \quad \text{and} \quad \sum_{i=1}^{n} \pi_i = 1. \tag{2.15}$$

This algorithm is based on the SR approach. In the forward stage of the GTH/S algorithm, we calculate a sequence of stochastic matrices $P_1, P_2, ...,$ and P_n, by eliminating one state from P in each iteration using Lemma 2.1. The elimination of a state at the j^{th} step, $1 < j \leq n$, corresponds to a transformation of the initial MC to a censored MC with transition matrix, P_j. The censored MC is the original MC observed only in the set that is not eliminated. As a result, every matrix, P_j, in this sequence has dimension one less than the previous matrix, P_{j-1}. After $n-1$ states are eliminated, the last matrix, P_n, has a single state with a trivial invariant

distribution, $\pi_n = \{1\}$. In the backward stage, a sequence of invariant distributions, π_i, for $i = n-1, \ldots, 2, 1$, is calculated in the reverse order.

$$P = P_1 \Longrightarrow P_2 \Longrightarrow \cdots P_s \cdots \Longrightarrow P_{n-1} \Longrightarrow P_n = \{1\}, \tag{2.16}$$

$$\pi = \pi_1 \Longleftarrow \pi_2 \Longleftarrow \cdots \pi_s \cdots \Longleftarrow \pi_{n-1} \Longleftarrow \pi_n = \{1\}.$$

We apply the GTH/S algorithm for modified transition matrices in different stages of our algorithm in Chapter 3. This algorithm involves approximately $\frac{2}{3}n^3 + 2n^2 - \frac{8}{3}$ operations. It has been shown to be more stable and accurate [21]. A pseudo-code presented by Sheskin [25] is given in the Appendix.

CHAPTER 3: THE IP MODEL: THE INVARIANT DISTRIBUTION

7 The Invariant Distribution π

Let $M = (S, P)$ be an IP model where $S = \sum_{i=1}^{k} L_i$, and $L_i = R_i + T_i$. Here, we use $+$ to denote a union of disjoint sets. Also, let T denote a disjoint union of all ports, $T = \sum_{j=1}^{k} T_j$, and $|T| = t$. We have also assumed that $t \leq$ (the size of the biggest island). For simplicity, we assume that $t = |L_i| = m, 1 \leq i \leq k$.

Our main goal is to present the IP algorithm which is used to compute π for the IP model. The main advantage of this algorithm, due to the unique features of the state space, is that most of the calculations solving for π are performed in *parallel*. Note that the possibility of parallel calculations in our model is not obvious. The invariant distribution on each island depends on transition probabilities on all of the other islands, even when transitions between islands are weak.

The invariant distribution, π, for any ergodic MC, with a $n \times n$ transition matrix P, is calculated by solving the linear system

$$\pi^T = \pi^T P \quad \Longleftrightarrow \quad \pi^T (I - P) = 0, \tag{3.1}$$

where $\pi^T = [\pi_1 \ \pi_2 \ \ldots \pi_n]$ satisfying $\sum_{i=1}^{n} \pi_i = 1, \ \pi_i > 0$, and I is a $n \times n$ identity matrix. There are many direct and iterative methods to solve (3.1). Some direct methods include the GTH/S algorithm, the LU factorization (with some adjustment), and iterative methods include the Gauss-Seidel, the Jacobi method, the power method,

and other hybrid schemes (see e.g. [33, 23]). Algorithms to calculate π for nearly uncoupled MCs include iterative schemes such the KMS algorithm in [16] and the KMS-GTH/S hybrid algorithm in [6].

8 SR Approach for the IP Model

We now present the recursive formula used in the backward stage of the GTH/S algorithm. We state this formula in the form given as Lemma 1 in [27].

Lemma 3.1. *Let $M_1 = (S_1, P_1)$ be an ergodic Markov model, $S_2 = S_1 \setminus \{z\}$, and $M_2 = (S_2, P_2)$ be z-reduced Markov model. Let S_2 and state z communicate in model M_1, i.e., there are states $i, j \in S_2$ such that $p_1(i, z) > 0$ and $p_1(z, j) > 0$. If the invariant distribution $\pi_2(\cdot)$ exists in model M_2, the invariant distribution $\pi_1(\cdot)$ also exists in M_1 and can be calculated by formulas*

$$(i)\ \pi_1(y) = \alpha_1 \pi_2(y),\ \textit{for all}\ \ y \in S_2, \quad \textit{and} \quad \alpha_1 = 1 - \pi_1(z), \tag{3.2}$$

$$(ii)\ \pi_1(z) = \beta_1 \sum_{y \in S_2} \pi_2(y) p_1(y, z) = \beta_1 R_1, \tag{3.3}$$

where $\alpha_1 = \frac{1}{1+n_1 R_1}$, $n_1 \equiv n_1(z) = \frac{1}{1-p_1(z,z)}$ and $R_1 \equiv R_1(z) = \sum_{y \in S_2} \pi_2(y) p_1(y, z)$.

We make some useful remarks based on our results so far.

Remark 3.2. Both Lemma 2.5 and Lemma 3.1 imply that invariant distributions of states in the reduced model are proportional to their distributions in the original model. However, Lemma 3.1 gives an explicit recursive formula that relates any two consecutive distributions in (2.16). This is particularly useful for calculations.

Remark 3.3. Instead of a single state $\{z\}$, now suppose we eliminate a set $D \subset S_1$ from model $M_1 = (S_1, P_1)$ to obtain model $M_2 = (S_2, P_2)$, where $S_2 = S_1 \setminus D$

and P_2 is given by formula (2.2). Lemma 3.1 implies that $\pi_1(y) = e\pi_2(y)$ for all $y \in S_2$ and for some scalar $e > 0$. Similarly, if $L \subseteq S_2$, then $\pi_1(L) = e\pi_2(L)$, where $\pi_i(L) = \sum_{x \in L} \pi_i(x), i = 1, 2$. We can use Lemma 2.5 to show that $e = \sum_{x \in S_2} \pi_1(x)$.

We will apply Remark 3.3 in the extension step of Stage 3 of our algorithm.

Remark 3.4. Suppose that we know the invariant distribution π_s in (2.16) for some stochastic matrix $P_s, 1 \leq s \leq k$, calculated after $s-1$ iterations of the initial matrix P. Using Lemma 3.1, we can also recursively calculate the invariant distributions $\pi_j,\ 1 \leq j < s$, in the backward stage of the GTH/S algorithm starting with the distribution π_s.

The following lemma follows directly from Lemma 2.1.

Lemma 3.5. *Let Markov model $M_1 = (S_1, P_1)$ be given. Suppose $S_2 = S_1 \setminus \{z\}$, and let $M_2 = (S_2, P_2)$ be the z-reduced model obtained from model M_1 where P_2 is obtained by formula (2.4). Now suppose $p_1(i, z) = 0$ or $p_1(z, j) = 0$ for $i, j \in S_2$, then $p_2(i, j) = p_1(i, j)$ in model M_2, i.e. elimination of state z does not change the probabilities between states i and j in M_2.*

Proof. The proof follows from formula (2.4) applied to $x = i, y = j$, and $z = z$. □

Remark 3.6. Suppose $D \subset S_1$ and one of $p_1(i, z) = 0$ or $p_1(z, j) = 0$ for all $z \in D$, then repeated application of Lemma 3.5 for states $z \in D$ results in $p_D(i, j) = p_1(i, j)$ for such states $j \in S_1 \setminus D$ in model M_D. That is, set D can also be eliminated without affecting the probabilities for state $i \in S_2$ and some states $j \in S_2$.

Lemma 3.5 and Remark 3.6 will play an important role in Stages 1 and 3 of the IP algorithm in the next section.

9 Main Results: The IP Algorithm

The IP algorithm consists of *three main stages*. The *third stage* involves two steps. Without going into detail, we first outline *the three stages of the IP algorithm*:

(*Stage 1*) In the first stage, we apply the SR approach to eliminate the interior states, R_i, from each island $L_i, 1 \leq i \leq k$. By Lemma 3.5 and Remark 3.6, this can be done in parallel for each island L_i. The elimination of interior states, $\bigcup_{i=1}^{k} R_i$, results in the model, $M^* = (T, P^*)$, for the ports T; see Figure 2a.

(*Stage 2*) In the second stage, we calculate the invariant distribution, π^*, for model M^*. Any direct method can be used for this calculation. We use the GTH/S algorithm to calculate π^* in our numerical example; see Figure 2b.

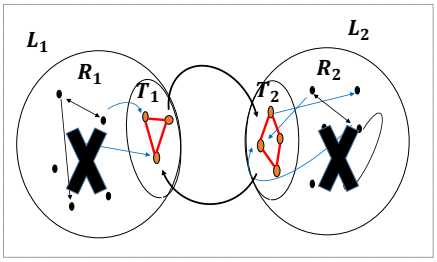
(*Stage 3. step 1.*) In the third stage, we consider models, $M_i = (L_i^*, P_i^*)$, for *augmented islands*, $L_i^* = R_i \cup T, 1 \leq i \leq k$. Using the distribution π^* obtained in the second stage, we calculate the invariant distribution , $\pi_{(i)}$, for each model M_i separately, and in parallel; see Figure 2c.

Remark 3.7. Note that the notation $\pi_{(j)}$ only refers to the invariant distribution of model M_j and not that of the individual states in M_j. For example, $\pi_{(j)}(x)$ refers to the invariant distribution of state $x \in L_j^*$.

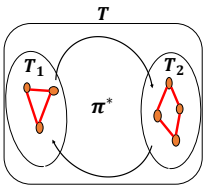
For each model M_i, we calculate the coefficients, w_i and λ_i, which are given by

$$w_i = \pi_{(i)}(R_i) \quad \text{and} \quad \lambda_i = \frac{w_i}{1 - w_i}, \quad i = 1, 2, \ldots, k, \tag{3.4}$$

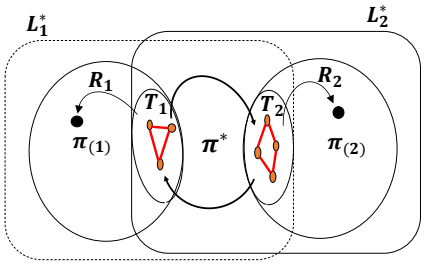
where $\pi_{(i)}(R_i) = \sum_{y \in R_i} \pi_{(i)}(y)$.



(a) *Stage 1.* Elimination of R_1 and R_2.



(b) *Stage 2.* Model M^* with invariant distribution π^*.



(c) *Stage 3. step 1.* Calculation of $\pi_{(1)}$ and $\pi_{(2)}$ for models M_1 and M_2, respectively.

Figure 2: Summary of the three stages of the IP Algorithm for an example when $k = 2$.

(*Stage 3. step 2. (Extension step)*) Finally, we use the distributions, $\pi_{(i)}, 1 \leq i \leq k$, to calculate the distribution π for model M. According to Remark 3.3, the invariant distribution of L_i inside model $M_i, 1 \leq i \leq k$, is proportional to its distribution w.r.t. model M, that is,

$$\pi(L_i) = e_i \pi_{(i)}(L_i), \qquad i = 1, 2, \ldots, k, \tag{3.5}$$

where $e_i = \pi(L_i^*) = \sum_{y \in L_i^*} \pi(y)$. If the coefficients e_i were known, we could

directly use formula (3.5) to calculate distribution $\pi(L_i)$ in each model $M_i, 1 \leq i \leq k$, but they are not known. Our main result, Theorem 1, gives an explicit formula (3.7) to calculate these unknown coefficients $e_i, 1 \leq i \leq k$.

Theorem 1. *The invariant distribution of each state $y \in L_i^*$ is given by the formula*

$$\pi(y) = e_i \pi_{(i)}(y), \quad i = 1, 2, \ldots, k, \tag{3.6}$$

where coefficients e_i are given by

$$e_i = \frac{1 + \lambda_i}{1 + \sum_j \lambda_j}, \qquad i = 1, 2, \ldots, k. \tag{3.7}$$

We give a proof of Theorem 1 at the end of this section.

Remark 3.8. Since $T = \bigcup_{i=1}^{k} T_i$, formula (3.6) in Theorem 1 implies that any one of the islands could be used to find the distribution on T because $e_1\pi_{(1)}(x) = e_2\pi_{(2)}(x) = \cdots = e_k\pi_{(k)}(x)$ for all $x \in T$. These equalities follow from (3.5) written for L_i^*. It is easy to check that $\pi = [\pi(1)\ \pi(2) \ldots \pi(s) \ldots \pi(n)], s \in S$, is a probability distribution.

We now discuss each stages of the IP algorithm in greater detail.

Stage 1

Consider an IP model $M = (S,\ P)$ with the state space $S = \sum_{i=1}^{k} L_i$ and transition matrix P. Let us first represent the stochastic matrix, P, as a union of blocks by *ignoring the blocks that are always zero.* Here, by a union of blocks we mean a collection of submatrices of a matrix. We will use such a representation for our matrices to make our discussion easy to follow. Figure 3a shows the representation

for P in (3.8) below.

$$P = \Big(\bigcup_{i=1}^{k} P_i\Big) \cup \Big(\bigcup_{\substack{i,j=1 \\ i\neq j}}^{k} T_{ij}\Big), \quad P_i = \begin{bmatrix} P_{i_0 i_0} & P_{i_0 i} \\ P_{i i_0} & P_{ii} \end{bmatrix}, \quad P^* = \Big(\bigcup_{i=1}^{k} P^*_{ii}\Big) \cup \Big(\bigcup_{\substack{i,j=1 \\ i\neq j}}^{k} T_{ij}\Big). \tag{3.8}$$

In (3.8) above, the block $P_i := \{p(x,y),\ x,y \in L_i\}$ and the block $T_{ij} := \{p(x,y),\ x \in T_i, y \in T_j, i \neq j\}$. As in (2.1), we decompose each block P_i as the second matrix in (3.8), where submatrices $P_{i_0 i_0} := \{p(x,y),\ x,y \in R_i\}, P_{i_0 i} := \{p(x,y),\ x \in R_i, y \in T_i\}, P_{ii} := \{p(x,y),\ x,y \in T_i\}$, and $P_{i i_0} := \{p(x,y),\ x \in T_i, y \in R_i\}$.

Now let $R = \sum_{i=1}^{k} R_i$ be the collection of all interior states, $T = \sum_{i=1}^{k} T_i$ be the collection of all ports of model M, and $S = R + T$. In stage 1, we eliminate R from S to obtain a reduced model $M^* = (T, P^*)$ for the ports T. The matrix P^* can also be represented as the third matrix in (3.8) where $P^*_{ii} := \{p^*(x,y),\ x,y \in T_i\}$ is the block with new probabilities for the ports, T_i calculated using formula (2.2) from the block P_i after R_i is eliminated. It is given by

$$P^*_{ii} = P_{ii} + P_{i i_0} N_{i_0} P_{i_0 i}, \quad i = 1, 2, \ldots, k. \tag{3.9}$$

We have used the notation '$*$' in block P_{ii} of matrix P^* in (3.8) to indicate that the elimination of $R = \bigcup_{i=1}^{k} R_i$ affects only the blocks P_{ii} but blocks T_{ij} are not affected for $i \neq j, 1 \leq i, j \leq k$. According to Lemma 3.9, blocks $P^*_{ii}, 1 \leq i \leq k$, in matrix P^*, can be calculated in parallel.

Lemma 3.9. *The block P^*_{ii} for each island L_i can be calculated in parallel from block $P_i, 1 \leq i \leq k$,.*

Proof. First, recall that we mentioned earlier that the IP model is ergodic. This

implies that there are states $x \in R_i$ and $y \in T_i$ such that $p(x,y) > 0$ in each island $L_i, 1 \leq i \leq k$. Then according to Lemma 2.1 from Chapter 2, eliminating R from S affects probabilities of the remaining states in the ports T, $T_i \subset L_i, 1 \leq i \leq k$. Since $L_i \cap L_j = \emptyset$ and $p(x,y) = 0$ for all $x \in L_i$ and $y \in R_j$, $i \neq j, 1 \leq j \leq k$, Lemma 3.5 and Remark 3.6 from section 8 imply that the elimination of interior states, R_j, from islands $L_j, j \neq i, 1 \leq j \leq k$, has no effect on the probabilities of states in island L_i. As a result, only the elimination of R_i affects the block P_{ii} in P_i in (3.8). Thus the elimination of R can be done by eliminating R_i from the block P_i for island L_i, separately and in parallel. This is illustrated in Figure 3b. □

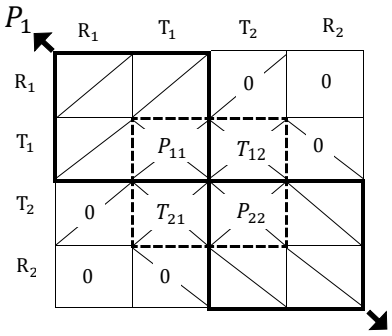
However, blocks T_{ij} remain unchanged because interior states in each island, L_i, do not communicate with the ports, T_j, from other islands, i.e. $p(i,j) = 0$ for $i \in R_i, j \in T_j, j \neq i, 1 \leq j \leq k$. By aggregating P^*_{ii} and T_{ij} for $i \neq j, 1 \leq i,j \leq k$, we obtain the matrix P^*. This is illustrated in Figure 3c for the case when $k = 2$.

Stage 2

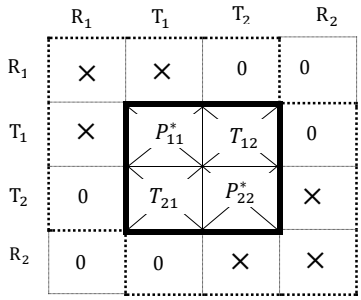
Let us consider a model $M^* = (T, P^*)$ for the ports T. The transition matrix P^* was obtained at the end of stage 1. In stage 2, we calculate the invariant distribution, π^*, for model M^*. For this calculation, we can use the GTH/S algorithm or any other direct method. Applying the GTH/S algorithm (as we do in our numerical example) has the complexity of order $\frac{2}{3}m^3$, where $|T| = |L_i| = m, 1 \leq i \leq k$.

Stage 3

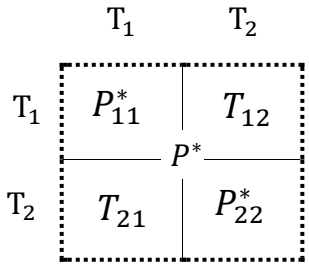
(Step 1) In this stage, we first consider "*augmented islands*" $L^*_i, 1 \leq i \leq k$, which are obtained by enlarging each island L_i to include all ports, i.e. $L^*_i = R_i \cup T$. Let



(a) Transition matrix P.



(b) Parallel elimination of R_1 and R_2 in blocks P_1 and P_2.



(c) P^* for model M^*.

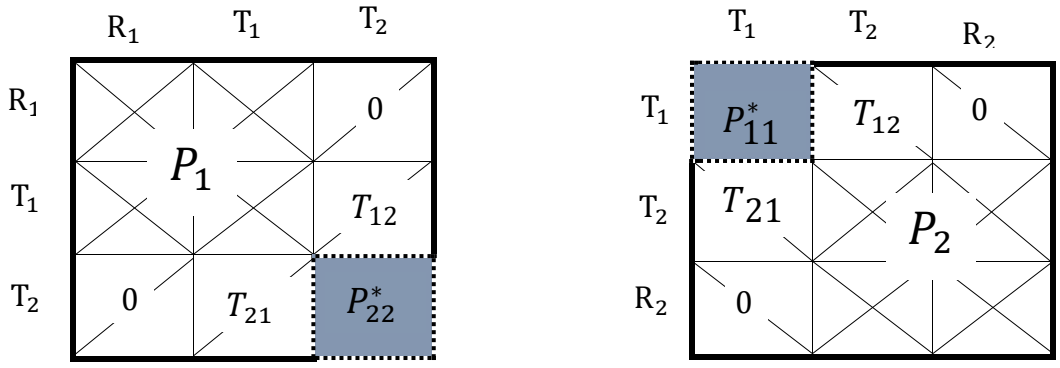
Figure 3: Summary of Stage 1 of the IP Algorithm: State reduction and calculation of P^* for the case $k = 2$.

us define new *models*, $M_i = (L_i^*, P_i^*), 1 \leq i \leq k$, with state space L_i^* and transition matrix P_i^*. Although the state space L_i^* of each model $M_i, 1 \leq i \leq k$, includes T, i.e. $T = \bigcap_{i=1}^{k} L_i^*$, each M_i is a separate model with its own transition matrix P_i^*. The matrix P_i^* for model M_i^* *can be calculated* from the initial matrix P by eliminating all states outside of L_i^*. That is, we eliminate $\bigcup_{j=1}^{k} R_j, j \neq i$ from S. However, let us show that the matrix P_i^* can be obtained from Stage 1 without any additional calculation. Indeed, by ignoring the parts that are always zero, the matrix P_i^* can

also be represented as

$$P_i^* = P_i \cup \Big(\bigcup_{\substack{j=1 \\ j \neq i}}^{k} P_{jj}^* \Big) \cup \Big(\bigcup_{\substack{i,j=1 \\ i \neq j}}^{k} T_{ij} \Big), \qquad i = 1, 2, \ldots, k. \tag{3.10}$$

Blocks $P_{jj}^*, 1 \leq j \leq k, j \neq i$, were calculated in stage 1 and blocks P_i and T_{ij} can be obtained directly from the original matrix P. Figures (4a) and (4b) show how matrices P_1^* and P_2^* are obtained for case $k = 2$.



(a) P_1^* for model M_1.

(b) P_2^* for model M_2.

Figure 4: IP Algorithm: Transition matrices P_1 and P_2 for the case $k = 2$.

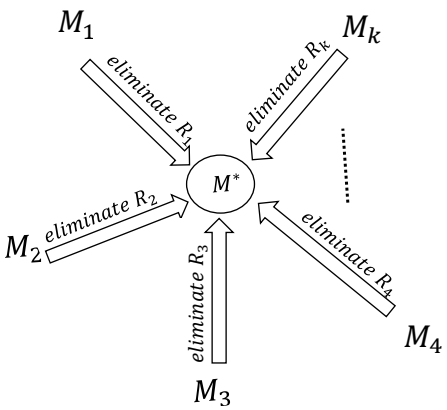
The block form in (3.10) of matrix P_i^* also follows directly from Lemma 3.5 and Remark 3.6 which implies that the elimination of the interior states, R_j, from matrix P in (3.8) only affects the block, P_{jj}, inside the block P_j of each island $L_j, 1 \leq j \leq k$. As we discussed in Stage 1, we indicate this change by updating P_{jj} to P_{jj}^*.

According to Remark 3.4 in Section 8, the invariant distribution, π, in (2.16) can be calculated from the distribution, π_s, of an intermediate stochastic matrix, $P_s, 1 \leq s \leq n$. From the discussion in Chapter 1, the matrix, P_s, is the transition matrix of a reduced model, M_s, after eliminating states $\{1, 2, 3, \ldots, s-1\}$ from the initial model $M_1 = (S_1, P_1)$. This fact leads to Lemma 3.10.

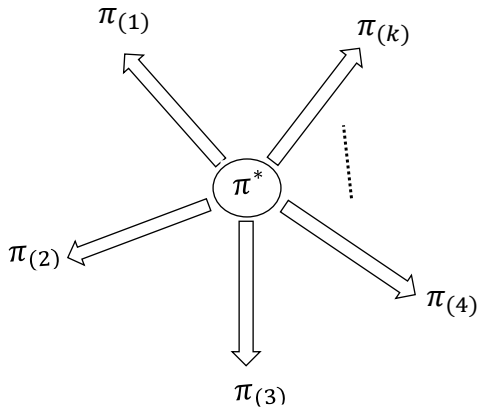
Lemma 3.10. *The calculation of the invariant distribution, $\pi_{(i)}$, of each model $M_i, 1 \leq i \leq k$, can be done in parallel.*

Proof. For each $i \in \{1, 2, \ldots, k\}$, let $M_i = (L_i^*, P_i^*)$ be the initial model. If we eliminate just the interior states, $R_i \subset L_i^* = L_i + T$, from M_i, we obtain a reduced model for the ports T only. The model we obtain is the model M^*, which was introduced in Stage 2. In other words, the model M^* is an R_i-*reduced* model of the model M_i. Because $T = \bigcap_{i=1}^{k} L_i^*$, the model M^* is also an R_i-reduced model of every model $M_i,\ 1 \leq i \leq k$. Given that the model M^* is a common model for each separate model $M_i, 1 \leq i \leq k$, we can start with the distribution, π^*, and use Lemma 3.1 to recursively compute $\pi_{(i)}$ for each model M_i in parallel. □

This concludes the discussion of the three stages of the IP algorithm. In Figure 5a, we summarize Stages 1 and 2 of the IP algorithm; in Figure 5b we illustrate Stage 3 (step 1) of the algorithm.



(a) Parallel elimination of interior states R_i from M_i to obtain a common model $M^*,\ 1 \leq i \leq k$.



(b) Parallel calculation of $\pi_{(i)}$ from common distribution $\pi^*,\ 1 \leq i \leq k$.

Figure 5: Parallel features of the IP algorithm.

(Step 2)In the extension step, we now use distributions $\pi_{(i)},\ 1 \leq i \leq k$, calculated

in Stage 3 (step 1), to compute the actual distribution, π, for model M using formulas (3.6) and (3.7) from Theorem 1.

Proof of Theorem 1: Let us first introduce the coefficients $r_i = \pi(R_i)$ and $t = \pi(T)$. Notice that proportionality in (3.5) also holds for the interior states $R_i \subset L_i^*$,

$$\pi(R_i) = e_i \pi_{(i)}(R_i) \quad i.e. \quad r_i = e_i w_i, \qquad i = 1, 2, \ldots, k. \tag{3.11}$$

Since $L_i^* = R_i \cup T$ and $S = \sum_{i=1}^{k} R_i \cup T$, we have the equalities

$$r_i = e_i - t, \tag{3.12}$$

$$\sum_{i=1}^{k} r_i + t = 1. \tag{3.13}$$

We substitute (3.12) into (3.13) to get $\Sigma_i(e_i - t) + t = \Sigma_i e_i - (k-1)t = 1$. Hence,

$$t = \frac{\sum_i e_i - 1}{k-1}. \tag{3.14}$$

We substitute (3.14) for t in (3.12) to obtain $r_i = e_i - (\sum_i e_i - 1)/(k-1)$. By the second equality in (3.11)

$$e_i - \frac{\sum_i e_i - 1}{k-1} = e_i w_i,$$

$$e_i(1 - w_i)(k-1) = \sum_i e_i - 1. \tag{3.15}$$

Let $e = \sum_i e_i$ and $\lambda_i = w_i/(1 - w_i)$, or *equivalently*, $1 - w_i = 1/(1 + \lambda_i)$. Thus, λ_i gives the ratio of the probability mass of interior states R_i to that of ports T in each model M_i, $1 \leq i \leq k$. Equation (3.15) can be simplified to

$$e_i = \frac{(e-1)(1+\lambda_i)}{k-1}. \tag{3.16}$$

If we sum (3.16) over all $i,\ 1 \leq i \leq k$, we obtain

$$e = \frac{e-1}{k-1} \sum_i (1+\lambda_i) = (e-1)\lambda, \tag{3.17}$$

where we let $\lambda = \sum_i (1+\lambda_i)/(k-1)$. Equation (3.17) can be rearranged to give $e - 1 = 1/(\lambda - 1)$. Then, we can simplify (3.16) as

$$e_i = \frac{(1+\lambda_i)}{(\lambda-1)(k-1)}. \tag{3.18}$$

We use the expression for λ to obtain

$$(k-1)(\lambda-1) = (k-1)\left(\frac{\sum_i (1+\lambda_i)}{k-1} - 1\right) = (k-1)\left(\frac{k+\sum_i \lambda_i}{k-1} - 1\right) = 1 + \sum_i \lambda_i. \tag{3.19}$$

Substitution of the expression (3.19) in the equality (3.18) gives

$$e_i = \frac{1+\lambda_i}{(k-1)(\lambda-1)} = \frac{1+\lambda_i}{1+\sum_i \lambda_i}, \tag{3.20}$$

with $\lambda_i = w_i/(1-w_i)$. The expression in (3.6) follows by applying formula (3.5) to the augmented island L_i^*.

□

We include a pseudo-code in the Appendix (2).

10 Performance Evaluation

Let us assume that there are n states, k islands, and n/k states in each island. If we let m be the number of ports, then there are $(n/k - m)$ interior states in each island.

(Stage 1) In stage 1, we eliminate $(n/k - m)$ states from each island $L_i, 1 \leq i \leq k$.

For each L_i, calculation of sequences of matrices, $W_{R_i}^{(i)}$, requires approximately $(n/k-m)((n/k)^2+1)$ additions and subtractions, $(n/k-m)((n/k)^2+1)$ divisions and multiplications; for a total of k islands, it requires approximately $2k(n/k-m)((n/k)^2+1)$ operations.

(Stage 2) For $T = km$ ports, the GTH/S algorithm requires $2/3(km)^3+2(km)^2-8/3$ operations while computing π for model M^*.

(Stage 3) (i) The calculation of each matrix $\hat{W}^{(i)}$, when restoring the interior states R_i for each model $M_i, 1 \leq i \leq k$, requires approximately the same number of operations as in the calculation of matrix $W_{R_i}^{(i)}$ in Stage 1. (ii) Each step in the calculation of $\pi_{(i)}$ using formulas (3.2)and (3.3) involves approximately $(n/k-m)$ additions, $(n/k-m)$ multiplications, 1 subtraction, and 2 divisions for each model. This results in a total count of $2k(n/k-m)^2+3(n/k-m)$ after $n/k-m$ states are restored. (iii) The calculation of w_i in (3.4) involves performing $n/k-m$ additions; calculation of λ_i requires 1 division and 1 subtraction for each island. (iv) Finally, the calculation of $e_i, 1 \leq i \leq k$, in (3.7) requires $3k+2$ additions and k divisions; calculation of π involves n multiplications.

We assume that the total number of ports $t \simeq n/k$ (size of the biggest island). If we suppose that there are an equal number of ports, m, in each L_i, i.e. $m = n/k^2$, then an approximate total count for the IP algorithm is given by

$$\textbf{Total count} \simeq \left(4k + \frac{2}{3}\right)\left(\frac{n}{k}\right)^3 \simeq \frac{n^3}{k^2} = mn^2.$$

In fact, since calculations are performed in parallel, we only need to consider the time

complexity for one island, thus, the time complexity of the IP algorithm is of the order $(\frac{n}{k})^3$.

11 Numerical Example

We consider a Markov model with the state space $S = \{1, 2, \ldots, 12\}$ and consisting of three islands: $L_1 = \{T_1 = (1,2)\} \cup \{R_1 = (3,4)\}$, $L_2 = \{T_2 = (5,6)\} \cup \{R_2 = (7,8)\}$, and $L_3 = \{T_3 = (9,10)\} \cup \{R_3 = (11,12)\}$. Further we assume that based on this partition of S, the transition matrix P can be rearranged as in Table 1. All calculations were performed in MATLAB with a tolerance of 10^{-7}.

Table 1: Transition matrix P.

$\left(\frac{1}{100}\right) \times$

	R_1		T						R_2		R_3	
			T_1		T_2		T_3					
	(3)	(4)	(1)	(2)	(5)	(6)	(9)	(10)	(7)	(8)	(11)	(12)
(3)	10	9	48	33	0	0	0	0	0	0	0	0
(4)	20	16	11	53	0	0	0	0	0	0	0	0
(1)	20	17	7	10	9	5	20	12	0	0	0	0
(2)	11	20	13	2	7	15	11	21	0	0	0	0
(5)	0	0	20	4	7	8	23	10	18	10	0	0
(6)	0	0	3	10	20	14	7	13	16	17	0	0
(9)	0	0	33	10	10	30	2	5	0	0	10	0
(10)	0	0	7	13	25	15	5	13	0	0	10	12
(7)	0	0	0	0	30	20	0	0	3	47	0	0
(8)	0	0	0	0	43	10	0	0	26	21	0	0
(11)	0	0	0	0	0	0	30	10	0	0	40	20
(12)	0	0	0	0	0	0	10	40	0	0	30	20

(Stage 1) We eliminate the interior states $R_1 = \{3, 4\}$ from L_1, states $R_2 = \{7, 8\}$ from L_2, and states $R_3 = \{11, 12\}$ from L_3 in parallel. After this elimination, the states that remain in L_1, L_2, and L_3 are ports $T_1 = \{1, 2\}, T_2 = \{3, 4\}$, and $T_3 = \{11, 12\}$, respectively. This elimination changes blocks P_{11} to P_{11}^*, P_{22} to P_{22}^*,

and P_{33} to P_{33}^* but blocks $T_{ij}, i \neq j, i, j = 1, 2, 3$, are unchanged. By aggregating these blocks, we obtain the stochastic matrix P^*. The matrix P^* is given as Table 2.

Table 2: Transition matrix P^*.

	R_1		T						R_2		R_3	
	(3)	(4)	(1)	(2)	(5)	(6)	(9)	(10)	(7)	(8)	(11)	(12)
(3)	0	0	0	0	0	0	0	0	0	0	0	0
(4)	0	0	0	0	0	0	0	0	0	0	0	0
(1)	0	0	640/2821	2578/8233	9/100	5/100	20/100	12/100	0	0	0	0
(2)	0	0	843/3449	725/3363	7/100	15/100	11/100	21/100	0	0	0	0
(5)	0	0	20/100	4/100	747/2771	471/2936	23/100	10/100	0	0	0	0
(6)	0	0	3/100	10/100	281/639	341/1481	7/100	13/100	0	0	0	0
(9)	0	0	33/100	10/100	10/100	30/100	43/525	37/420	0	0	0	0
(10)	0	0	7/100	13/100	25/100	15/100	13/84	103/420	0	0	0	0
(7)	0	0	0	0	0	0	0	0	0	0	0	0
(8)	0	0	0	0	0	0	0	0	0	0	0	0
(11)	0	0	0	0	0	0	0	0	0	0	0	0
(12)	0	0	0	0	0	0	0	0	0	0	0	0

(Stage 2) The invariant distribution, π^*, for model $M^* = (T, P^*)$ is given by

$$\pi^* = \begin{matrix} (1) & (2) & (5) & (6) & (9) & (10) \end{matrix}$$
$$\left[\begin{array}{cccccc} \frac{438}{2393}, & \frac{183}{1237}, & \frac{317}{1522}, & \frac{544}{3209}, & \frac{359}{2442}, & \frac{440}{3051} \end{array} \right].$$

Here, we used the GTH/S algorithm for this calculation.

(Stage 3) We consider three augmented islands: $L_1^* = \{1, 2, 3, 4, 5, 6, 9, 10\}$, $L_2^* = \{1, 2, 5, 6, 7, 8, 9, 10\}$, and $L_3^* = \{1, 2, 5, 6, 9, 10, 11, 12\}$. For these, we consider three Markov models $M_1 = (L_1^*, P_1^*)$, $M_2 = (L_2^*, P_2^*)$, and $M_3 = (L_3^*, P_3^*)$. Matrices P_1^*, P_2^*, and P_3^* are given in Table 3 for models M_1, M_2, and M_3, respectively.

Using formulas (3.2) and (3.3) in Lemma 3.1, we compute the invariant distribution

$\pi_{(1)}$ for model M_1 *(only states* $1, 2, 3, 4$*)* which are given by

$$\pi_{(1)} = \begin{matrix} (1) & (2) & (3) & (4) \end{matrix} \left[\begin{matrix} \frac{320}{2023}, & \frac{213}{1666}, & \frac{387}{5843}, & \frac{119}{1711} \end{matrix} \right].$$

The distributions $\pi_{(2)}$ and $\pi_{(3)}$ for models M_1 *(only states* $5, 6, 7, 8$*)* and M_2 *(only states* $9, 10, 11, 12$*)* are calculated in a similar way, which gives

$$\pi_{(2)} = \begin{matrix} (5) & (6) & (7) & (8) \end{matrix} \left[\begin{matrix} \frac{300}{1759}, & \frac{265}{1909}, & \frac{431}{5301}, & \frac{119}{1192} \end{matrix} \right], \ \pi_{(3)} = \begin{matrix} (9) & (10) & (11) & (12) \end{matrix} \left[\begin{matrix} \frac{306}{2303}, & \frac{186}{1427}, & \frac{254}{4143}, & \frac{79}{2265} \end{matrix} \right].$$

Using formula (3.4), we calculate the coefficients $w = (w_1, w_2, w_3)$ and $\lambda = (\lambda_1, \lambda_2, \lambda_3)$ for models M_1, M_2, and M_3, which are given by

$$w = \left(\frac{691}{5089}, \frac{121}{668}, \frac{618}{6425} \right), \quad \lambda = \left(\frac{691}{4398}, \frac{121}{547}, \frac{618}{5807} \right).$$

We use formula (3.7) in Theorem 1 to calculate the coefficients $e = (e_1, e_2, e_3)$

$$e = \left(\frac{611}{784}, \frac{1177}{1431}, \frac{3372}{4525} \right).$$

We can verify that the above coefficients are correct by calculating $e_i = \pi(R_i)/\pi_{(i)}(R_i)$.

Finally, we calculate π using formula (3.6),

$$\pi = \begin{matrix} (1) & (2) & (3) & (4) & (5) & (6) & (7) & (8) & (9) & (10) & (11) & (12) \end{matrix} \left[\begin{matrix} \frac{438}{3553}, & \frac{138}{1385}, & \frac{67}{1298}, & \frac{325}{5996}; & \frac{171}{1219}, & \frac{269}{2356}, & \frac{495}{7402}, & \frac{325}{3958}; & \frac{241}{2434} & \frac{281}{2893} & \frac{197}{4312} & \frac{137}{5271} \end{matrix} \right].$$

Table 3: Transition matrices for models M_1, M_2, and M_3.

	R_1		T					
	(3)	(4)	(1)	(2)	(5)	(6)	(9)	(10)
(3)	10/100	9/100	48/100	33/100	0	0	0	0
(4)	20/100	16/100	11/100	53/100	0	0	0	0
(1)	20/100	17/100	7/100	10/100	9/100	5/100	20/100	12/100
(2)	11/100	20/100	13/100	2/100	7/100	15/100	11/100	21/100
(5)	0	0	20/100	4/100	747/2771	471/2936	23/100	10/100
(6)	0	0	3/100	10/100	281/639	341/1481	7/100	13/100
(9)	0	0	33/100	10/100	10/100	30/100	43/525	37/420
(10)	0	0	7/100	13/100	25/100	15/100	13/84	103/420

(a) Transition matrix P_1^*.

	T						R_2	
	(1)	(2)	(5)	(6)	(7)	(8)	(9)	(10)
(1)	640/2821	2578/8233	9/100	5/100	0	0	20/100	12/100
(2)	843/3449	725/3363	7/100	15/100	0	0	11/100	21/100
(5)	20/100	4/100	7/100	8/100	18/100	10/100	23/100	10/100
(6)	3/100	10/100	20/100	14/100	16/100	17/100	7/100	13/100
(7)	0	0	30/100	20/100	3/100	47/100	0	0
(8)	0	0	43/100	10/100	26/100	21/100	0	0
(9)	33/100	10/100	10/100	30/100	0	0	43/525	37/420
(10)	7/100	13/100	25/100	15/100	0	0	13/84	103/420

(b) Transition matrix P_2^*.

	T						R_3	
	(1)	(2)	(5)	(6)	(9)	(10)	(11)	(12)
(1)	640/2821	2578/8233	9/100	5/100	20/100	12/100	0	0
(2)	843/3449	725/3363	7/100	15/100	11/100	21/100	0	0
(5)	20/100	4/100	747/2771	471/2936	23/100	10/100	0	0
(6)	3/100	10/100	281/639	341/1481	7/100	13/100	0	0
(9)	33/100	10/100	10/100	30/100	2/100	5/100	10/100	0
(10)	7/100	13/100	25/100	15/100	5/100	13/100	10/100	12/100
(11)	0	0	0	0	30/100	10/100	40/100	20/100
(12)	0	0	0	0	10/100	40/100	30/100	20/100

(c) Transition matrix P_3^*.

12 Applications: Nearly Uncoupled MCs; Perturbation Estimates

As we mentioned in Chapter 1, transitions between states in the IP model are more restrictive than that in the nearly uncoupled (ergodic) Markov model. Most of the applications discussed in Chapter 1 are examples of the nearly uncoupled MCs. In this section, we will discuss the possibilities of considering the IP model as an approximation of a nearly uncoupled Markov model. We will show that such an approximation can be understood in the sense of *perturbation* applied to a stochastic matrix.

Assume that $M_1 = (S_1, P_1)$ is a nearly uncoupled model. Let $S_1 = \sum_{i=1}^{k} L_i, |S_1| = n$, and $L_i = R_i + T_i$. For simplicity, suppose that most of the transitions between clusters L_i and $L_j, i \neq j$, occur only through some states in T_i, but assume that there are also small transitions between some states $i \in R_i$ and $j \in R_j, i \neq j, 1 \leq i, j \leq k$, i.e. $0 < p_1(i,j) \leq \epsilon$, for some ϵ close to zero. In this simple case, we can obtain another model, $M_2 = (S_2, P_2)$, from model M_1 by redistributing these small probabilities, $p_1(i,j), i \neq j, i \in R_i, j \in R_j$, back to each island $L_i, 1 \leq i \leq k$. Then, M_2 can be thought of as a new model resulting after some *perturbation* is *applied* to the original model M_1. Most importantly, M_2 is now an IP model. Before we discuss the perturbation theory of stochastic matrices and present some known results, we investigate, for this example, the following three ways to redistribute small probabilities in M_1:

(*a*) Let $\epsilon_i = \sum_j p_1(i,j)$. Then we can divert probabilities $p_1(i,j), i \in R_i, j \in \{R_1, R_2, \ldots, R_j\}, i \neq j$, back to the state of origin, i.e. $p_2(i,i) = p_1(i,i) + \epsilon_i$.

All other probabilities remain the same. Now there are no small transitions between the islands other than through state i.

(*b*) Let $|T_i| = t_i$ be the number of ports in the island L_i. We can also distribute ϵ_i evenly to the ports T_i, i.e. $p_2(i,j) = p_1(i,j) + \epsilon_i/t_i, j \in T_i$.

(*c*) Similar to (*b*) but we can also redistribute ϵ_i evenly to all states in $L_i, 1 \leq i \leq k$.

Let π_1 and π_2 be the invariant distributions for models M_1 and M_2 respectively. Let us define the total relative error as

$$\text{Total relative error} = \sum_{i=1}^{n} \left| \frac{\pi_1(i) - \pi_2(i)}{\pi_1(i)} \right| \times 100 \tag{3.21}$$

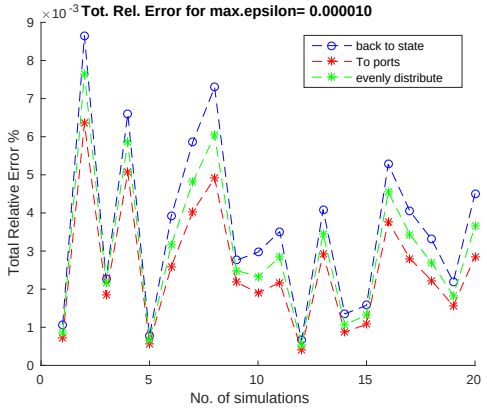
We performed a simple simulation for an example with 2 clusters to examine if (*a*), (*b*) or (*c*) is the best method for the redistribution of small probabilities. Note that there many other ways of such redistribution in addition to the three ways we propose. Our example consists of clusters, $L_1 = \{1,2,3,4\}$ and $L_2 = \{5,6,7,8,9\}$, with most interactions occurring through states $T_1 = \{3,4\}$ and $T_2 = \{5,6\}$ respectively. All calculations were performed using the GTH/S algorithm.

Simulation 1. Consider state $1 \to 7$ in the model M_1 with a randomly generated stochastic matrix P_1[3]. Let $p_1(1,7) = \epsilon$, where $\epsilon \in \text{uniform}[0, \epsilon_{max}]$. All other transitions between L_1 and L_2 occur through T_1 and T_2. For the model M_2, we tested method (*a*) where we divert ϵ back to $p_1(1,1)$, i.e. $p_2(1,1) = p_1(1,1) + \epsilon$. We calculate π_2 and compute the total relative error; then we tested method (*b*) where we let $p_2(1,j) = p_1(1,j) + \epsilon/2, j \in T_1$; finally in method (*c*), $p_2(i,j) = p_1(i,j) + \epsilon/4$, if

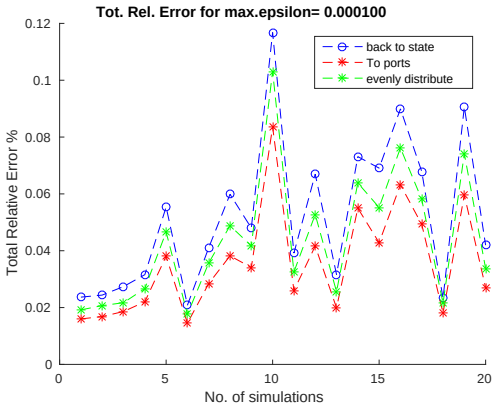
[3]For each simulation we generate a new matrix P_1

$i, j \in L_1$. By varying ϵ_{max}, we plotted the total relative errors for 20 simulations[4].

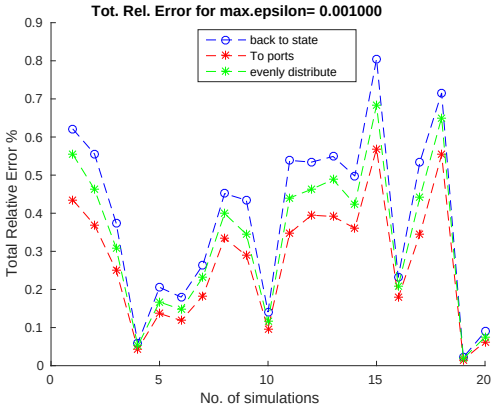
For the case of one small transition, as we increase ϵ_{max} in Figures 6a–6d, we see that the total relative error increases as expected, but method (b) gives the lowest total relative error.



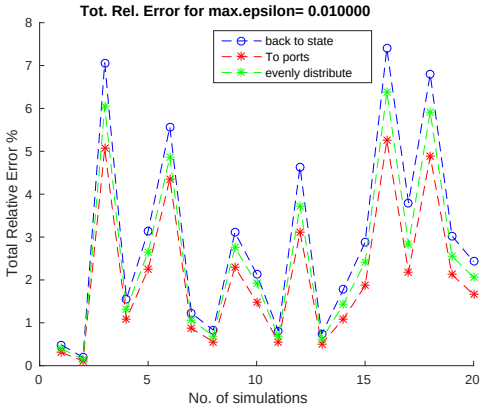
(a) $\epsilon = \text{unif}[0, 0.00001]$.



(b) $\epsilon = \text{unif}[0, 0.0001]$.



(c) $\epsilon = \text{unif}[0, 0.001]$.



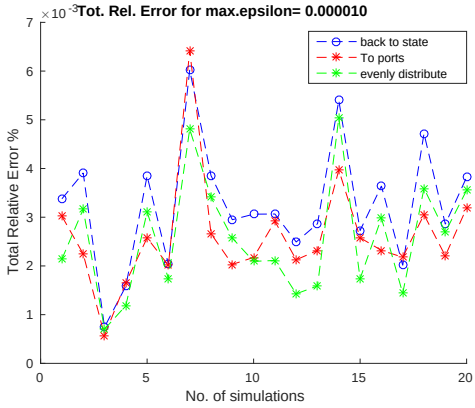
(d) $\epsilon = \text{unif}[0, 0.01]$.

Figure 6: An example of redistribution of small probabilities (1 state case).

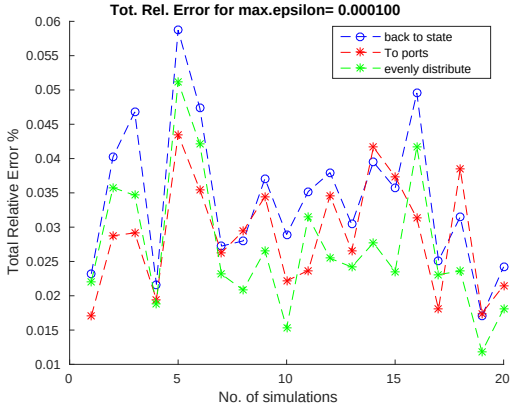
Simulation 2. We now consider the case when $p_1(1, 7) = \epsilon_1, p_1(8, 2) = \epsilon_2$, where

[4]Pattern is similar for $N = 100$ or larger. We chose $N = 20$ so that it is easy to see the pattern.

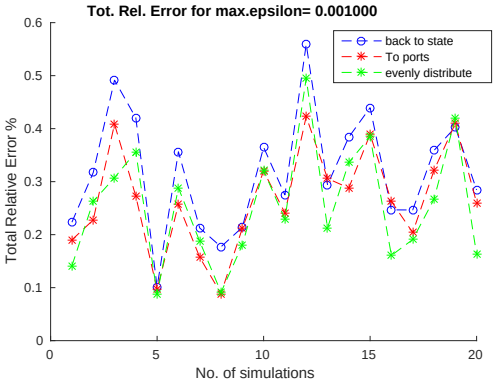
$\epsilon_1, \epsilon_2 \in \text{unif}[0, \epsilon_{max}]$. We calculate the total relative error using method (a) where $p_2(1,1) = p_1(1,1) + \epsilon_1$, $p_2(8,8) = p_1(8,8) + \epsilon_2$; and we repeat with (b) where $p_2(1,j) = p_1(1,j) + \epsilon_1/2, j \in T_1$ and $p_2(8,j) = p_1(8,j) + \epsilon_2/2, j \in T_2$; in method (c) $p_2(i,j) = p_1(i,j) + \epsilon_1/4$, if $i, j \in L_1$ and $p_2(i,j) = p_1(i,j) + \epsilon_2/5$, if $i, j \in L_2$.



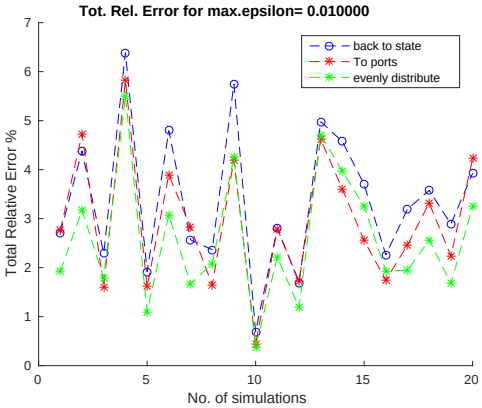
(a) $\epsilon = \text{unif}[0, 0.00001]$.



(b) $\epsilon = \text{unif}[0, 0.0001]$.



(c) $\epsilon = \text{unif}[0, 0.001]$.



(d) $\epsilon = \text{unif}[0, 0.01]$.

Figure 7: An example of redistribution of small probabilities (2 states case).

In Figure 7a we can see that the total relative error is small when ϵ_1 and ϵ_2 are small. For the case with 2 states, we can see that redistributing small transitions

evenly among all states in each island consistently gives the lowest relative error. Of course, this must be tested for large state spaces with more states participating in such transitions. From Figures 6 and 7, we can see that the total relative error is about 6% if $\epsilon_1, \epsilon_2 \in \text{unif}[0, 0.01]$. For small transitions, the IP model may not be a bad approximation of the original model. We may choose to use method (b) or (c) for the redistribution of small probabilities.

Perturbation Estimates

In this section, we give a brief survey of the perturbation theory of stochastic matrices. Although we do not present any new results, we use existing results to understand the redistribution of small transitions, which was discussed in the previous section. In Chapter 1, we also mentioned that the effect of perturbation on the invariant distributions of stochastic matrices have been rigorously studied in the literature. This is particularly relevant to our discussion. First, we mention the two types of perturbations that have been mostly studied:

Linear perturbation: A perturbation of this type is of the form $P_2 = P_1 + E$, where E is a matrix such that each row sum, $\sum_j e(i,j) = 0$ and the norm $||E||$ is small relative to 1. Both matrices P_1 and P_2 are stochastic. The matrix P_2 is also referred to as the *perturbed* matrix.

Analytic perturbation: A matrix P_0 is said to be analytically perturbed if the perturbation is in the form of power series

$$P(\epsilon) = P_0 + \epsilon P_1 + \epsilon^2 P_2 + \ldots, \tag{3.22}$$

where the coefficient matrices $P_k, k = 1, 2, \ldots,$ are known and $\epsilon \in [0, \epsilon_{max}]$. The above series is assumed to converge in some non-empty neighborhood of $\epsilon = 0$. If the matrix coefficients, $P_2 = P_3 = \cdots = 0$, then the linear perturbation is a special case of the analytical perturbation in (3.22). Various numerical methods to solve perturbation of type (3.22) can be found in [3]. Our discussion only involves the case of linear perturbation.

Let us consider the case of linear perturbation, $P_2 = P_1 + E$, where P_2 and P_1 are stochastic matrices, and E is some small perturbation applied to P_1. Let π_1 and π_2 be the invariant distributions for matrices P_1 and P_2, respectively. Then, we have the following norm-wise perturbation bounds

$$||\pi_2 - \pi_1|| \leq k||E||, \tag{3.23}$$

where k is a condition number. The number k is different under different norms. A comparison of all the condition numbers is given in [5]. We state the bound given in [4]. This bound states that sensitivity to perturbation can be measured using mean passage times.

Theorem 2. *[4]*

$$||\pi_2 - \pi_1||_\infty \leq \frac{1}{2} \max_j \left[\frac{\max_{i \neq j} \ m_{ij}}{r_j} \right] ||\Delta P||_\infty \tag{3.24}$$

where $m_{ij} = \frac{v_{jj} - v_{ij}}{\pi_j}$, $r(j) = \frac{1}{\pi_j}$, $\Delta P = P_2 - P_1$, *and* V *is the fundamental matrix of matrix* P_1.

Notice that when we redistribute small probabilities back to each island according to method (b) or (c), we form another stochastic matrix that is slightly perturbed. We

may use the bound given by Theorem 2 to estimate a bound for our approximation. Let P_2 be obtained from P_1 by directing all small transitions of the interior states according to (b) or (c). For both the cases, it is easy to see that $\Delta P = P_2 - P_1$, row sums of $\Delta P = 0$, and $||\Delta P||_\infty \leq 2\epsilon_{max}$. Here, $\epsilon_{max} = \max\{\epsilon_1, \epsilon_2, \ldots, \epsilon_i, \ldots, \epsilon_k\}$ and $\epsilon_i = \sum_j p_1(i,j), i \neq j, i \in R_i, j \in R_j, 1 \leq i, j \leq k$.

Then, the bound in (3.24) gives

$$||\pi_2 - \pi_1||_\infty \leq \epsilon_{max} \max_j \left[\frac{\max_{i \neq j} \ m_{ij}}{r_j} \right]. \tag{3.25}$$

There is also an explicit formula to calculate π_1, in terms of π_2, using the famous theorem by Schweitzer (1968)[24].

Theorem 3. *[24] Let (Z_n^1) and (Z_n^2) be two irreducible MCs defined on the same state space S with transition matrices P_1 and $P_2 = P_1 + E$, where E is some perturbation. Suppose π_1 and π_2 are the invariant distributions of MCs (Z_n^1) and (Z_n^2), respectively. And let V_1 be the fundamental matrix of P_1 (defined in (3.28)). Then*

$$\pi_2^T = \pi_1^T (I - EV_1)^{-1} \quad \textit{and} \quad \pi_2^T - \pi_1^T = \pi_2^T E V_1. \tag{3.26}$$

It is also possible to give another bound to the error given by (3.26).

Corollary 3.11. *[24] Under the conditions of Theorem 3,*

$$||\pi_2 - \pi_1||_1 \leq ||E||_\infty ||V_1||_\infty. \tag{3.27}$$

Proof. Using $\pi_2^T - \pi_1^T = \pi_2^T E V_1$, we can see that $||\pi_2 - \pi_1||_1 = ||(\pi_2^T E V_1)^T||_1 \leq ||V_1^T||_1 ||E^T||_1 = ||V||_\infty ||E||_\infty$. □

In the context of our discussion, for small $E = \Delta P$, it may be possible to use the

invariant distribution, π_2, of the perturbed model, M_2, which is also an IP model, to approximate the actual distribution, π_1, of model M_1. Note that this bound still requires the calculation of the fundamental matrix, Z_1, for an ergodic MC (Z_n^1) specified by model M_1. The fundamental matrix, Z_1, is given by

$$V_1 = (I - P_1 - A)^{-1}, \tag{3.28}$$

where $I = |P_1| \times |P_1|$ identity matrix, $A = \lim_{n\to\infty} P^n = e\pi_1^T$ is the limiting matrix of P and e is vector of size $|P_1| \times 1$. We may also use the algorithm given in [30] to calculate V_1. Even though we present an algorithm to calculate matrix, N, which is a fundamental matrix of a substochastic matrix, it could be interesting to investigate if we can calculate V_1 for the IP model, in parallel.

CHAPTER 4: THE IP MODEL: THE FUNDAMENTAL MATRIX

13 Fundamental Matrix N

Let $M = (S, P)$ be a homogeneous Markov model, where S is discrete, finite, and P is the transition matrix. Let (Z_n) be a MC specified by the model M with an initial distribution μ_0. If the MC (Z_n) is ergodic (irreducible, aperiodic, and positive recurrent), then the fundamental matrix V is defined as

$$V = (I - P - A)^{-1}, \tag{4.1}$$

where $I = |P_1| \times |P_1|$ is an identity matrix and $A = \lim_{n\to\infty} P^n = e\pi^T$ is the limiting matrix of P. The entries of $V := \{n(x, y) : x, y \in S\}$ give the expected number of visits of the MC (Z_n) to a state y from a state x. A recursive algorithm to compute V, based on the SR approach, is given in [30].

In some cases, one might be interested in calculating the expected number of visits of the MC (Z_n) to states inside some non-empty and non-absorbing subset $D \subseteq S$ before the time of the first exit to $S \setminus D$. Let $C = S \setminus D$ and $D = \{1, 2, \ldots, k\}$ be the enumeration of states in D. Let $Z_0 \in D$ and define the first passage time $\tau = \min\{n > 0 : Z_n \in C\}$.

Definition 1. The expected number of visits of the MC (Z_n) to $y \in D$ from $x \in D$

before the time of the first exit τ to C is given by

$$n(x,y) := E\Big[\sum_{n=0}^{\tau-1} \mathbb{1}_{\{Z_n=y\}} | Z_0 = x\Big] = \sum_{n=0}^{\infty} (Q)^n_{xy}, \qquad x,y \in D. \tag{4.2}$$

In matrix form, equation (4.2) can be written as

$$N = \sum_{n=0}^{\infty} Q^n, \tag{4.3}$$

where $Q := \{p(x,y) : x,y \in D\}$ is the transition probability matrix for states in D. The matrix N is also referred to as the fundamental matrix of the substochastic matrix Q.

Lemma 4.1. *The fundamental matrix N satisfies the equalities*

$$(a)\ N = I + QN \quad (b)\ N = I + NQ. \tag{4.4}$$

According to (4.4) we have

$$N = (I-Q)^{-1} \overset{(by\ 4.3)}{=} \sum_{n=0}^{\infty} Q^n. \tag{4.5}$$

The inverse and the sum in (4.5) exist. Probabilistically, C can be thought of as an absorbing state and the MC (Z_n), which gets absorbed in C at time τ, is then a transient MC. As a result, each state $y \in D$ is visited only finitely many times. Hence the sum in (4.5) converges and the middle equality follows from (4.4). Using results from linear algebra, it can also be shown that the spectral radius, σ, of any substochastic matrix Q is less than 1, i.e. $\sigma(Q) < 1$, and this implies that $\sum_{n=0}^{\infty} Q^n$ converges and is equal to $(I-Q)^{-1}$ (see e.g. [23]).

Lemma 4.2. *The distribution of the MC (Z_n) at the time of the first exit τ to C is*

given by $U = NT$.

(a) Initial model $M_1 = (S_1, P_1)$. (b) Censored model $M_2 = (S_2, P_2)$.

Figure 8: Censored MCs.

Let (Z_n^1) be a MC specified by model $M_1 = (S_1, P_1)$. And let $D \subseteq S_1$, $G \subseteq S_1$, $G \cap D = \emptyset$, and $S_2 = S_1 \setminus D$. A sample path of the original MC, (Z_n^1), is shown in Figure 8a. Let $\tau_0 = 0, Z_0^1 \in S_2 \setminus G$, and $\tau_{n+1} = \min\{n > \tau_n : Z_n^1 \in S_2\}$. By Lemma 2.1, $(Z_n^2) = (Z_{\tau_n}^1)$ is a censored MC specified by the reduced-model $M_2 = (S_2, P_2)$. The sample path for the reduced MC,(Z_n^2), is shown in Figure 8b. Notice that the visits to D, which was eliminated, are censored.

Define the Markov times $\tau^{(i)} := \min\{n : Z_n^i \in G\}$ for $i = 1, 2$. Let the matrix $N_1 := \{n_1(x, y) : x, y \in S_1 \setminus G\}$ be the fundamental matrix of MC (Z_n^1) and $N_2 := \{n_2(x, y) : x, y \in S_2 \setminus G\}$ be the fundamental matrix of the censored MC (Z_n^2). Then the entries of matrices N_1 and N_2 give the expected number of visits inside $S_1 \setminus G$ and $S_2 \setminus G$ by MCs (Z_n^1) and (Z_n^2) before the time $\tau^{(1)}$ and $\tau^{(2)}$ to set G, respectively. By Lemma 4.2, the distributions at time $\tau^{(i)}$ for MCs (Z_n^i) are given by $U_i = N_i T_i,\ i = 1, 2$, where $T_i = \{p_i(x, y),\ x \in S_i \setminus G, y \in G\}$.

The following Proposition is due to Sonin [31]. It states that the characteristics

given by matrices N and U are invariant under censoring.

Proposition 4.3. *[31] Let $M_1 = (S_1, P_1)$ be a Markov model. And let $G \subset S_1, G \cap D = \emptyset, D \subset S_1$, and $S_2 = S_1 \setminus D$. Let $M_2 = (S_2, P_2)$ be a D-reduced Markov model. Let N_1 and N_2 be the fundamental matrices for MCs (Z_n^1) and (Z_n^2) defined for $S_1 \setminus G$ and $S_2 \setminus G$, respectively. Then the elements of matrices U_1, N_1 of model M_1 restricted to model M_2 coincide with the corresponding elements of matrices U_2, N_2, i.e., $u_1^G(x,y) = u_2^G(x,y),\ x \in S_2 \setminus G,\ y \in G$ and $n_1^G(x,y) = n_2^G(x,y),\ x,y \in S_2 \setminus G$.*

According to Proposition 4.3, the distributions $u_2^G(x,y), y \in G$, of MC (Z_n^2) at the time of the first visit to G and the expected number of visits to y from x given by $n_2^G(x,y)$ are the same as that of MC (Z_n^1) for those states x and y that remain in model M_2. This property holds true for any finite number of repeated eliminations, provided that state, x, remains in the state space in the calculation of $u^G(x,y), y \in G$. Let $D = \{1, 2, \ldots, k-1, k\}$ be the enumeration of states in D. If we eliminate states $1, 2, \ldots, k-1$ from D, then for state k, Proposition 4.3 implies

$$u_1^G(k,y) = u_2^G(k,y) = \cdots u_{k-1} k^G(k,y) = p_k(k,y)/s_k, \quad y \in G,$$

$$n_1^G(k,k) = n_2^G(k,k) = \cdots n_{k-1}^G(k,k) = n_k = 1/\left(1 - p_k(k,k)\right).$$

Our goal in this chapter is to develop the FUNDQ algorithm to compute the fundamental matrix N. We will apply the SR approach, use the identities given by (4.4), and the results of Proposition 4.3 to develop this algorithm.

14 The FUNDQ Algorithm

Let $M = (S, P)$ be a model and suppose P is a $r \times r$ transition matrix. Let $M_1 = (S_1, Q_1)$ be the substochastic model, where $S_1 \subset S$ is a non-absorbing set, $n < r$, and $Q_1 = \{p(x, y) : x, y \in S_1\} \subset P$ is a $n \times n$ substochastic matrix for S_1. Let N_1 be the fundamental matrix of matrix Q_1. There is an algorithm presented by [12] to compute N_1, which was also developed in the framework of the SR approach . This algorithm has the time complexity of order n^3. In comparison, the FUNDQ algorithm, we develop in the next two sections, has the time complexity of order $k(n^2) + (n - k)^3, 1 \leq k < n$. Because this algorithm is a recursive algorithm, it consists of two stages — a forward stage and a backward stage. In the forward stage, we apply the SR approach on Q_1 to eliminate states from S_1 to obtain a smaller substochastic matrix $Q_s, 1 \leq s < n$. We calculate the fundamental matrix, N_s, for this matrix Q_s. In the backward stage, starting with the matrix, N_s, we recursively compute a sequence of fundamental matrices by inserting states that were previously eliminated during the forward stage, and calculate the matrix N_1 for Q_1.

We now describe these two stages of the algorithm in detail.

Stage 1. Forward Stage

In this stage, we apply Lemma 2.1 Chapter 2 on the substochastic matrix Q_1, instead of the stochastic matrix P_1, to eliminate states $s \in S_1$, $1 \leq s < n$, in $n - 1$ iterations, to obtain a matrix Q_n

$$Q_1 \Longrightarrow Q_2 \cdots \Longrightarrow Q_k \cdots \Longrightarrow Q_n = \{p_n(n, n)\}, \tag{4.6}$$

where the matrix $Q_n = \{n\}$ has only one state (also see Remark 4.4). We calculate the fundamental matrix, N_n, for Q_n, which is given by

$$N_n = \frac{1}{1 - p_n(n,n)}.$$

Remark 4.4. Lemma 2.1 Chapter 2 gives formula (2.2) to calculate the transition matrix for a censored MC. However, formula (2.2) can also be applied to a substochastic matrix Q. Let $D_0 \subseteq S_1$, then the matrix Q can also be decomposed as

$$Q = \begin{array}{c} \begin{matrix} \scriptstyle (D_0) & \scriptstyle (S_1 \setminus D_0) \end{matrix} \\ \begin{bmatrix} A & T \\ R & K \end{bmatrix} \end{array}.$$

We can calculate the matrix Q_0, which describes transitions inside of $S_1 \setminus D_0$, by eliminating D_0 using formula (2.2)

$$Q_0 = K + RN_{D_0}T,$$

where $N_{D_0} = (I - A)^{-1}$ is the fundamental matrix for the substochastic matrix A.

Stage 2. Backward Stage

In the backward stage of the FUNDQ algorithm, we apply formulas given by Proposition 4.6 to recursively calculate a sequence of matrices

$$N_1 \Longleftarrow N_2 \cdots \Longleftarrow N_k \cdots \Longleftarrow N_n. \tag{4.7}$$

In each iteration, using Proposition 4.6, we insert a state $s \in S_1$ that was eliminated during the forward stage, and compute the sequence in (4.7). Thus, the matrix N_k in

(4.7) represents the fundamental matrix for previously inserted states $\{k+1, \ldots, n-1, n\}$, including the state k that is inserted at this step.

Remark 4.5. As in Remark 3.4 Chapter 3, it is also possible to eliminate states $\{1, 2, \ldots, k-1\} \subset S_1, k < n$, and compute matrix $N_k = (I - Q_k)^{-1}$. Then we can recursively calculate matrix N_1 starting from N_k in $k-1$ iterations.

Remark 4.5 will be useful in the algorithm to calculate the fundamental matrix for the IP model later in this chapter.

Note that Proposition 4.6 gives formulas for one-step recursive calculation of matrices in (4.7). The sequence in (4.7) can be calculated from the repeated application of Proposition 4.6 for each previously eliminated state $s \in S_1$.

Proposition 4.6. *Let $M = (S, P)$ be a Markov model and suppose $M_1 = (S_1, Q_1), S_1 \subseteq S, Q_1 \subseteq P$, be the substochastic model. Let N_1 be the fundamental matrix for Q_1. Assume, for simplicity, state $z \in S_1$ is eliminated from Q_1 and it is located in position (1,1), which we denote by A_1. Then the matrices Q_1 and N_1 can be represented as below:*

$$Q_1 = \begin{bmatrix} \boldsymbol{A}_1 & \boldsymbol{T}_1 \\ \boldsymbol{R}_1 & \boldsymbol{K}_1 \end{bmatrix}, \quad N_1 = \begin{bmatrix} \boldsymbol{u}_1 & \boldsymbol{n}_1 \\ \boldsymbol{m}_1 & \boldsymbol{N}_2 \end{bmatrix}. \tag{4.8}$$

The entries of matrix N_1 are given by

$$\boldsymbol{n}_1 = \frac{1}{s_z}(T_1 \cdot N_2), \ \boldsymbol{m}_1 = \frac{1}{s_z}(N_2 \cdot R_1), \ \boldsymbol{u}_1 = \frac{1}{s_z}(1 + n_1 \cdot R_1), \tag{4.9}$$

where $s_z = 1 - p_1(z, z)$ and $N_2 = (I - Q_2)^{-1}$ is the fundamental matrix for the

substochastic matrix $Q_2 = K_1 + \frac{1}{s_z}(R_1 \cdot T_1)$ *obtained from matrix* Q_1 *after eliminating* $\{z\}$.

We also give a pseudo-code (3) for the FUNDQ algorithm in the Appendix.

Proof of Proposition 4.6:

Suppose state $z \in S_1$ is eliminated. Then the row vector $\boldsymbol{n}_1$ can be obtained from the equality (4.4)(a)

$$n_1(z,y) = p_1(z,z)n_1(z,y) + \sum_{x \in S_1 \backslash \{z\}} p_1(z,x)n_1(x,y), \quad y \neq z, y \in S_1. \tag{4.10}$$

Proposition 4.3 implies that even after the elimination of state z, $n_1(x,y) = n_2(x,y)$ for all $x, y \in S_1 \backslash \{z\}$. We simplify (4.10) to obtain

$$n_1(z,y) = \frac{1}{s_z} \sum_{x \in S_1 \backslash \{z\}} p_1(z,x)n_2(x,y), \tag{4.11}$$

which, in matrix form, is $\boldsymbol{n}_1 = \frac{1}{s_z}(T_1 \cdot N_2)$.

From the equality (4.4) (b), the z^{th} column in matrix N_2 is given by

$$n_1(x,z) = \mathbb{1}_z(x) + n_1(x,z)p_1(z,z) + \sum_{y \in S_1 \backslash z} n_1(x,y)p_1(y,z), \ x \in S_1.$$

Solving for $n_1(x,z)$, we obtain

$$n_1(x,z) = \frac{1}{s_z}\Big(\mathbb{1}_z(x) + \sum_{y \in S_1 \backslash \{z\}} n_1(x,y)p_1(y,z)\Big). \tag{4.12}$$

By proposition (4.3) and noting $x \neq z$, $\mathbb{1}_z(x) = 0$, (4.12) can be written (we name it

$\mathbf{m}_1$) as

$$\mathbf{m}_1(x,z) = \frac{1}{s_z}\Big(\sum_{y\in D\setminus\{z\}} n_2(x,y)p_1(y,z)\Big).$$

In matrix form, this is $\boldsymbol{m}_1 = \frac{1}{s_z}(N_2 \cdot R_1)$. To calculate $\boldsymbol{u}_1$, we make use of the above results. If we replace z for x in equation (4.12)

$$u_1(z,z) = \frac{1}{s_z}\Big(1 + \sum_{y\in S_1\setminus\{z\}} n_1(z,y)p_1(y,z)\Big).$$

Replacing the equality for $n_1(z,y)$ from equation (4.11), we obtain

$$u_1(z,z) = \frac{1}{s_z}\Big[1 + \frac{1}{s_z}\sum_{y\in S_1\setminus\{z\}}\Big(\sum_{x\in S_1\setminus\{z\}} p_1(z,x)n_2(x,y)\Big)p_1(y,z)\Big]. \tag{4.13}$$

In matrix form, we can write (4.13) as $\boldsymbol{u}_1 = \frac{1}{s_z}\Big(1 + \frac{1}{s_z}\left((T_1 \cdot N_2)\cdot R_1\right)\Big) = \frac{1}{s_z}\left(1 + n_1 \cdot R_1\right)$, where we used the equality for $\boldsymbol{n}_1$ in the last equality.

□

15 Order Count for the FUNDQ Algorithm

Let Q_1 be $n \times n$ matrix. The calculation of N_1 using matrix inversion has the time complexity of order n^3. Let $D = \{1, 2, \ldots, k\} \subset S_1, k < n$, be the eliminated set (see Remark 4.5). In the FUNDQ algorithm,

(i) The calculation of W(and $\hat{W}$) matrices in k iterations requires approximately $2k(n^2+1)$ operations.

(ii) The calculation of the matrix N_{k+1} has the complexity of order $(n-k)^3$. In each step, the calculation of $\mathbf{n}$ and $\mathbf{m}$ vectors requires approximately $(n-k)^2$ multiplications, $(n-k)(n-k-1)$ additions, and 1 division. The calculation

of $\mathbf{u}$ requires about $(n-k)^2$ multiplications and $(n-k-1)$ additions; in k iterations, the calculation of $\mathbf{n}$, $\mathbf{m}$, and $\mathbf{u}$ requires $3k(n-k)^2+n$ operations.

The total count for our algorithm is approximately $4k(n^2+1)+(n-k)^3+3k(n-k)^2+n$. So, the time complexity is approximately of order $k(n^2)+(n-k)^3$.

16 Numerical Example

Consider a transition matrix P_1 given in Table 4. In this example, state 7 is absorbing. The submatrix in the square box is the matrix Q_1 and the matrix N_1 is calculated directly using the (first) formula in (4.5). All the numbers are rounded off to 4 decimal places.

Table 4: Matrices P_1 and N_1.

$P_1 =$

	[1]	[2]	[3]	[4]	[5]	[6]	[7]
[1]	0	0.4	0	0	0	0	**0.6**
[2]	0.6	0	0.4	0	0	0	**0**
[3]	0	0.6	0	0.4	0	0	**0**
[4]	0	0	0.6	0	0.4	0	**0**
[5]	0	0	0	0.6	0	0.4	**0**
[6]	0	0	0	0	0.6	0	**0.4**
[7]	**0**	**0**	**0**	**0**	**0**	**0**	**1**

, $N_1 =$

	[1]	[2]	[3]	[4]	[5]	[6]
[1]	1.6149	1.0248	0.6313	0.3691	0.1943	0.0777
[2]	1.5372	2.5619	1.5784	0.9228	0.4857	0.1943
[3]	1.4206	2.3677	2.9990	1.7533	0.9228	0.3691
[4]	1.2457	2.0763	2.6299	2.9990	1.5784	0.6314
[5]	0.9835	1.6391	2.0763	2.3677	2.5619	1.0248
[6]	0.5901	0.9835	1.2457	1.4206	1.5372	1.6149

We apply the FUNDQ algorithm for matrix Q_1. In the forward stage, we eliminate states $1, 3$ and 6. This results into the matrix Q_4 of size 3×3 which describes transitions for states $2, 4$, and 5. The fundamental matrix for Q_4 is given by $N_4 = (I - Q_4)^{-1}$.

Table 5: Fundamental matrix N_4 for Q_4.

$Q_4 =$

	[2]	[4]	[5]
[2]	0.48	0.16	0
[4]	0.36	0.24	0.40
[5]	0	0.60	0.24

$N_4 =$

	[2]	[4]	[5]
[2]	2.5619	0.9228	0.4857
[4]	2.0763	2.9990	1.5784
[5]	1.6391	2.3677	2.5619

In the backward stage, we insert states $3, 1$, and 6 (in this order) to obtain the matrices N_3, N_2, and N_1, shown in Table 6 respectively. We use a full-size 6×6 matrix (same size as the matrix Q_1) to show states that are restored.

Table 6: Insertion of states 3, 1 and 6.

N_4

	[1]	[2]	[3]	[4]	[5]	[6]
[1]	0	0	0	0	0	0
[2]	0	2.5619	0	0.9228	0.4857	0
[3]	0	0	0	0	0	0
[4]	0	2.0763	0	2.9990	1.5784	0
[5]	0	1.6391	0	2.3677	2.5619	0
[6]	0	0	0	0	0	0

N_3

	[1]	[2]	[3]	[4]	[5]	[6]
[1]	0	0	0	0	0	0
[2]	0	2.5619	1.5784	0.9228	0.4857	0
[3]	0	2.3677	2.9990	1.7533	0.9228	0
[4]	0	2.0763	2.6299	2.9990	1.5784	0
[5]	0	1.6391	2.0763	2.3677	2.5619	0
[6]	0	0	0	0	0	0

N_2

	[1]	[2]	[3]	[4]	[5]	[6]
[1]	1.6149	1.0248	0.6314	0.3691	0.1943	0
[2]	1.5372	2.5619	1.5784	0.9228	0.4857	0
[3]	1.4206	2.3677	2.9990	1.7533	0.9228	0
[4]	1.2457	2.0763	2.6299	2.9990	1.5784	0
[5]	0.9835	1.6391	2.0763	2.3677	2.5619	0
[6]	0	0	0	0	0	0

N_1

	[1]	[2]	[3]	[4]	[5]	[6]
[1]	1.6149	1.0248	0.6314	0.3691	0.1943	0.0777
[2]	1.5372	2.5619	1.5784	0.9228	0.4857	0.1943
[3]	1.4206	2.3677	2.9990	1.7533	0.9228	0.3691
[4]	1.2457	2.0763	2.6299	2.9990	1.5784	0.6314
[5]	0.9835	1.6391	2.0763	2.3677	2.5619	1.0248
[6]	0.5901	0.9835	1.2457	1.4206	1.5372	1.6149

17 Application: The Fundamental Matrix for the IP Model

(a) The fundamental matrices obtained from W matrices

In this section, we discuss the fundamental matrix, N, given by the equality (4.4) in the context of the IP model. Note that a fundamental matrix, N, appears in the block form (2.1) of the full matrix W in Chapter 2. Also recall that (see Remark 2.3 in Chapter 2) we perform all our calculations with the W matrices. To see why this is relevant to our discussion, recall that in Stage 1 of the IP algorithm in Chapter 3, we eliminated the interior states, R_i, from each island $L_i, 1 \leq i \leq k$. To elaborate on this further, let matrix $W^{(i)}, 1 \leq i \leq k$, denote the full matrix corresponding to

the i^{th} island. To calculate the block P^*_{ii} in matrix P^* in (3.8), we let $W^{(i)}_1 = P_i$ and eliminate states in R_i using formula (2.8) to obtain matrix $W^{(i)}_{R_i}$. As in (2.1), we can represent the matrix $W^{(i)}_{R_i}$ as

$$W^{(i)}_{R_i} = \begin{bmatrix} W^{(i)}_{11} & W^{(i)}_{12} \\ W^{(i)}_{21} & W^{(i)}_{22} \end{bmatrix} = \begin{bmatrix} P_{i_0 i_0} N_i & N_i P_{i_0 i} \\ P_{i i_0} N_i & P^*_{ii} \end{bmatrix}, \tag{4.14}$$

where the submatrices $P_{i_0 i_0}, P_{i_0 i}, P_{i i_0}$ and P^*_{ii}, were defined in the second matrix in (3.8). Then, the block P^*_{ii} is obtained from the matrix $W^{(i)}_{R_i}$ in (4.14), i.e. $W^{(i)}_{22} = P^*_{ii}$. All the other blocks P^*_{jj} are *calculated* from the matrices $W^{(j)}_{R_j}, 1 \le j \le k$, in a similar manner.

Let $\tau^i = \min(n > 0,\ Z_n \in T_i), 1 \le i \le k$, denote the first passage time for some MC (Z_n), specified by the IP model M, to T_i from some state in $R_i, 1 \le i \le k$. And let us introduce the fundamental matrices $N_i := \{n_i(x, y) : x, y \in R_i\}, 1 \le i \le k$. For each $i = 1, 2, \ldots, k$, the entries of N_i give the expected number of visits of MC (Z_n) to the interior states, R_i, before the time, τ^i, to the ports, T_i, in island L_i. To obtain N_i from $W^{(i)}_{R_i}$, we recall that the equality (4.4) in Chapter 3 implies that the matrix N_i satisfies

$$N_i = I + P_{i_0 i_0} N_i = I + W^{(i)}_{11}, \tag{4.15}$$

where I is an $|R_i| \times |R_i|$ identity matrix. Using the submatrix $W^{(i)}_{11}$ from matrix $W^{(i)}_{R_i}$ in (4.14), we can calculate N_i using formula (4.15) for $R_i, 1 \le i \le k$. That is, for

$x, y \in R_i$,

$$n_i(x,y) = \begin{cases} 1 + W_{11}^{(i)}(x,x), & \text{if } x = y, \\ W_{11}^{(i)}(x,y), & \text{otherwise.} \end{cases}$$

Most importantly, we do not require any additional calculations to compute $N_i, 1 \leq i \leq k$, because they can be directly obtained from the W matrices, as we described above, which were obtained when we calculated π in Chapter 3.

(b) The IP Fund Algorithm

In this section, we present another algorithm to calculate a fundamental matrix for the case we formulate below. For this, we will make use of Proposition 4.6 of the FUNDQ algorithm.

Let $M = (S, P)$ be an IP model. Let D_i be the set of non-absorbing states in each island $L_i, 1 \leq i \leq k$. Let $Q \subset P$ be the substochastic matrix for states in $S \setminus D$, where $D = \bigcup_{i=1}^{k} D_i$. For simplicity, we assume that $D_i \subset R_i$ and $D_i \cap T_i = \emptyset$ for all $1 \leq i \leq k$. Let $C_i = R_i \setminus D_i$ and $S_i = L_i \setminus D_i = C_i + T_i$. We can easily visualize our formulation by considering the case when $k = 2$ as shown in Figure 9.

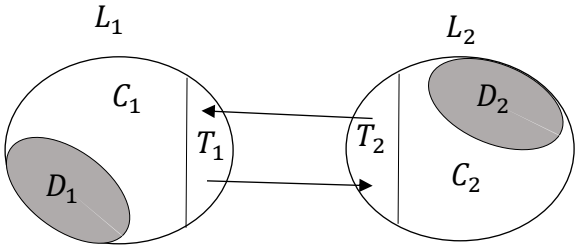


Figure 9: IP Fund Algorithm: Representation of the state space for the IP model as an example for the case when $k = 2$.

Following our representation of a matrix in (3.8) Chapter 3, the matrix Q may also

be represented as a union of blocks, *ignoring the parts of Q that are always zero.*

$$Q = \Big(\bigcup_{i=1}^{k} Q_i\Big) \cup \Big(\bigcup_{\substack{i,j=1 \\ i\neq j}}^{k} T_{ij}\Big), \quad i,j = 1,2,\ldots,k, \tag{4.16}$$

where $Q_i := \{p(x,y),\ x,y \in S_i\}$ and block $T_{ij} := \{p(x,y),\ x \in T_i, y \in T_j\}$. The fundamental matrix $N = (I-Q)^{-1}$ gives the expected number of visits inside $S \setminus D$ by some MC (Z_n) specified by the IP model, before the time of the first exit τ_0 to D, i.e. $\tau_0 = \min\{n > 0 : Z_n \in D\}, Z_0 \in S \setminus D$. A matrix representation for the case when $k = 2$ is also shown in Figure 10a.

The algorithm to calculate N for the case described above consists of three stages. We describe each stage in detail.

Stage 1

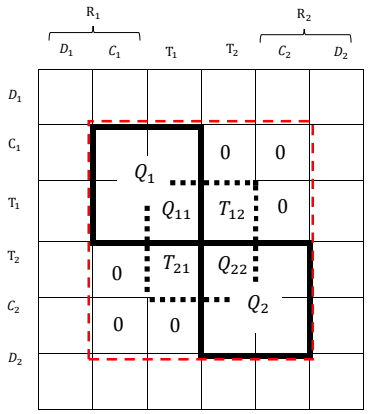
For each $S_i, 1 \leq i \leq k$, we eliminate C_i from the block, Q_i, to obtain a smaller block Q^*_{ii}. After $\bigcup_{i=1}^{k} C_i$ is eliminated from Q, that is, C_i have been eliminated from all $S_i, 1 \leq i \leq k$, we obtain the matrix Q^*. This matrix can also be represented as

$$Q^* = \Big(\bigcup_{i=1}^{k} Q^*_{ii}\Big) \cup \Big(\bigcup_{\substack{i,j=1 \\ i\neq j}}^{k} T_{ij}\Big), \tag{4.17}$$

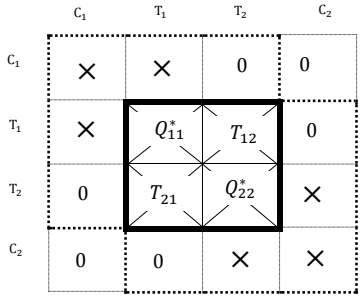
where $Q^*_{ii} := \{p^*(x,y) : x,y \in T_i\}$. The matrix, Q^*, is the substochastic matrix for the ports T.

Remark 4.7. According to Lemma 3.5 and Remark 3.6 in Chapter 3, the elimination of interior states in C_i only affects the block Q_i, which results in block Q^*_{ii}. Thus, we can also calculate each block $Q^*_{ii}, 1 \leq i \leq k$, in parallel. The substochastic matrix, Q^*, can be computed by aggregating block, Q^*_{ii}, and the unchanged block

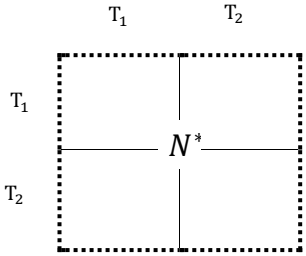
$T_{ij}, i \neq j, 1 \leq i, j \leq k$. See Figures 10a and 10b for an example for the case when $k = 2$.



(a) Substochastic matrix Q (in red), D_1 and D_2 are absorbing.



(b) Parallel elimination of blocks C_1 and C_2.



(c) Matrix N^*.

Figure 10: IP Fund Algorithm: Summary of Stages 1 and 2 of an example for the case when $k = 2$.

Stage 2

In this stage, using the substochastic matrix, Q, calculated in Stage 1 we calculate the fundamental matrix, $N^* = (I - Q^*)^{-1}$, for T. This can be done either using matrix inversion or by applying the FUNDQ algorithm to Q^* (see Figure (10c).

Stage 3

Let us represent the matrix N in terms of row-blocks

$$N = [F_1\ F_2\ \ldots\ F_i \ldots F_k]^T, \tag{4.18}$$

where each row block $F_i := [N(i,1) \ldots N(i,i)\ \ N(i,i+1) \ldots N(i,k)], 1 \leq i \leq k$, each block $N(i,i) := \{n(x,y) : x \in S_i, y \in C_i + T\}$, and $N(i,j) := \{n(x,y) : x \in S_i, y \in C_j\}, i \neq j, 1 \leq i,j \leq k$. We have illustrated this representation in Figure 11b.

We may also represent each row block F_i as

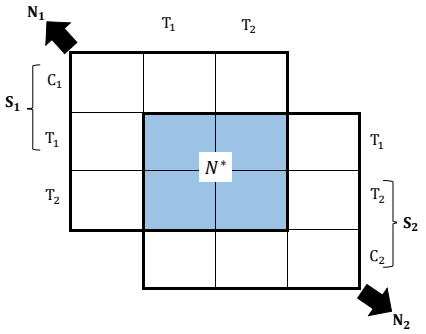
$$F_i = N(i,i) \cup \Big(\bigcup_{\substack{j=1 \\ i \neq j}}^{k} N(i,j) \Big), \qquad i = 1, 2, \ldots, k.$$

Now let us define the block $N_i := \{n(x,y) : x, y \in C_i + T\}$. Note that $N(ii) \subset N_i$. Because of the structure of the IP model and previous results, we have the following result.

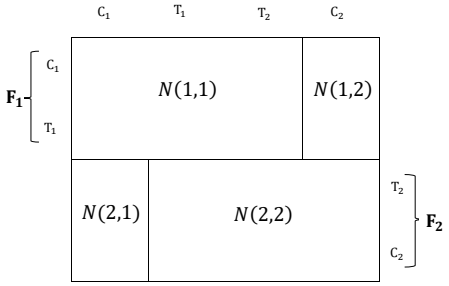
Lemma 4.8. *Each block $N_i, 1 \leq i \leq k$, can be computed in parallel.*

Proof. Because $N^* = \bigcap_{i=1}^{k} N_i$, the proof follows by replacing N_2 with N^* in Proposition 4.6 and recursively inserting eliminated states $x \in C_i$ for each island $L_i, 1 \leq i \leq k$ separately (also see Fig.(11a). The calculation of the block, N_i, does not depend on the calculation of the block, $N_j, j \neq i$, because states in C_i do not communicate with states in $C_j, 1 \leq j \leq k$. Since elimination can be done in parallel, insertion, which is a reverse operation, can also be done in parallel. □

We now discuss how blocks $N(i,j), i \neq j, 1 \leq i,j \leq k$, can be computed. Let us assume that the first state to be restored is the state $z \in C_j$. According to the



(a) Calculation of blocks N_1 and N_2 in parallel.



(b) Row block decomposition of matrix N.

Figure 11: IP Fund Algorithm: An example of the decomposition of matrix N for the case $k = 2$.

equality (4.4)(b), the expected number of visits from $x \in S_i$ to $z \in C_j, j \neq i$, is simply the expected number of visits to $j \in T_j$ from x, and then, transitioning from j to z with probability $p(j, y)$, i.e. $n(x, z) = \sum_{j \in T_j} n(x, j)p(j, z), x \in S_i$. After state z is restored, we repeat the same argument. That is, we insert $z' \in C_j$, then expected number of visits from $x \in S_i$ to $z' \in C_j$ is now the expected number of visits to $T_j \bigcup \{z\}$, and then transitioning to z'.

Now suppose blocks $N(i, i), 1 \leq i \leq k$, were computed using Lemma 4.8, then let block $N(T_j) := \{n(x, y) : x \in S_i, y \in T_j\}$ be the part of the block, $N(i, i)$, which gives the expected number of visits to the ports, T_j, from states in $S_i, i \neq j$. Also, let $z_1^{(j)}, z_2^{(j)}, .., z_{c_j}^{(j)}$, be the enumeration of states in $C_j, |C_j| = c_j$, in some (arbitrary) order of insertion. Then each column of the block $N(i, j)$ can be calculated in an iterative manner using Lemma 4.9.

Lemma 4.9. *Let $N_j^1 = N(T_j)$. Then the s^{th} column of block $N(i,j)$, given by $\boldsymbol{m}_s$, can be calculated for $s = 1, 2, 3, \ldots, c_j$,*

(i) $\boldsymbol{m}_s = \frac{1}{s_z}\left(N_j^s \cdot \boldsymbol{R}_s^{(j)}\right)$ *where* $s_z = 1 - p(z_s^{(j)}, z_s^{(j)})$,

(ii) $N_j^{s+1} = [N_j^s \quad \boldsymbol{m}_s]$,

where $\boldsymbol{R}_s^{(j)} = \{p(x, z_s) : x \in C_i \cup (z_1^{(j)}, z_2^{(j)}, \ldots, z_{s-1}^{(j)})\}$ *is the insertion probability vector of state $z_s^{(j)}$ state in $C_j \subset L_j$, i.e., when $\{z_{s+1}^{(j)}, z_{s+2}^{(j)}, .., z_{c_j}^{(j)}\}$ are still eliminated from C_j.*

Proof. The proof follows from Proposition 4.6 by repeating the calculation of vectors m_z for $z \in \{z_1^{(j)}, z_2^{(j)}, .., z_{c_j}^{(j)}\}$. □

In Lemma 4.9, we can see that the calculation of the block, $N(i,j)$, requires knowing the probability vectors, $\mathbf{R}_s^{(j)}$, for each inserted state $z_s^{(j)}, 1 \leq s \leq c_j$. In particular, this requires communication between all of the islands, i.e. probability vector $\mathbf{R}_s^{(j)}$ is the vector of state $z_s^{(j)}$ from set $C_j \subset L_j$ obtained after states, $z_1^{(j)}, z_2^{(j)}, .., z_{s-1}^{(j)}$(and $z_s^{(j)}, z_{s+1}^{(j)}, .., z_{c_j}^{(j)}$, yet to be insterted), have been restored from the block C_j. We suggest a way to obtain the vectors $\mathbf{R}_s^{(j)}$ without any further calculations. But this requires storing a total of $c_1 + c_2 + \cdots + c_j$ vectors from all the islands, and therefore, can take up lots of memory.

Let $z_1^{(i)}, z_2^{(i)}, .., z_{c_i}^{(i)}$ be the order of elimination of states from set $C_i \subset L_i, |C_i| = c_i, 1 \leq i \leq k$. After each state is eliminated from block Q_i in (4.16), we obtain a sequence of matrices

$$Q_i = Q_1^{(i)} \to Q_2^{(i)} \to \ldots Q_s^{(i)} \cdots \to Q_{c_i}^{(i)}, \quad i = 1, 2, \ldots, k. \tag{4.19}$$

If we store the vector, $R_s^{(i)}$, which is the vector corresponding to the state $z_s^{(i)} \in C_i, 1 \leq s \leq c_i$[5], while eliminating C_i from matrix Q_i in Stage 1, we obtain the sequence

$$\mathbf{R}_1^{(i)} \to \mathbf{R}_2^{(i)} \to \ldots \mathbf{R}_s^{(i)} \cdots \to \mathbf{R}_{c_i}^{(i)}, \quad i = 1, 2, \ldots, k.$$

Then, for each block $N(i,j), i \neq j$, we can simply insert the states in C_j in the same order as we eliminated them in Stage 1 using the probability vectors $\mathbf{R}_s^{(i)}, 1 \leq s \leq c_i$. In this way, we can now apply Lemma 4.9 to calculate the block $N(i,j), i \neq j$, by calculating the vectors

$$\mathbf{m}_1 \to \mathbf{m}_2 \cdots \to \mathbf{m}_{c_i}. \tag{4.20}$$

After calculating all of the blocks, $N(i,j), j \neq i, j = 1, 2, \ldots, k$, we aggregate these blocks, together with the block, $N(i,i)$, we then obtain the row block $F(i), i = 1, 2, \ldots, k$; and finally, we obtain the matrix N according to the representation in (4.18).

We give a pseudo-code (4) in the Appendix.

18 Numerical Example

Consider an IP model $M = (S, P)$ with the state space consisting of 14 states, i.e. $S = \{1, 2, 3, \ldots, 11, 12, 13, 14\}$, and the transition matrix P is given in Table 7. Let $L_1 = \{1, 2, 3, 4\}, L_2 = \{5, 6, 7, 8\}$, and $L_3 = \{9, 10, 11, 12, 13, 14\}$, be the three islands of this model with ports $T_1 = \{1, 2\}, T_2 = \{5, 6\}$, and $T_3 = \{9, 10\}$, respectively. Let $D = \{3, 13, 14\}$ where $D_1 = \{3\}, D_2 = \emptyset$, and $D_3 = \{13, 14\}$. Define the Markov

[5]It will be easier to see this if we let Q_i to be the matrix Q_1 in Proposition 4.6, then $R_s^{(i)}$ is the vector R_1 corresponding to state $z_s^{(i)}$, say, WLOG is z_1.

time $\tau = \min\{n > 0 : Z_n \in D\}$. Let $Z_0 \in S \setminus D$. In this case, we are interested in the expected number of visits to states in islands $R_1 \setminus D_1$, R_2, and $R_3 \setminus D_3$ by a MC before the time τ.

Table 7: Matrices P and Q (dotted in red).

$\frac{1}{100}$

			R_1		T					R_2		R_3			
					T_1		T_2		T_3						
			3	4	1	2	5	6	9	10	7	8	11	12	13	14
R_1	3	10	35	30	25	0	0	0	0	0	0	0	0	0	0
	4	20	26	10	44	0	0	0	0	0	0	0	0	0	0
T_1	1	20	10	10	10	10	10	20	20	0	0	0	0	0	0
	2	23	16	15	4	7	20	10	5	0	0	0	0	0	0
T_2	5	0	0	7	13	10	13	19	10	18	10	0	0	0	0
	6	0	0	20	10	7	24	5	10	13	11	0	0	0	0
T_3	9	0	0	31	10	10	8	8	9	0	0	10	3	5	6
	10	0	0	3	18	17	5	5	11	0	0	19	12	2	8
R_2	7	0	0	0	0	27	11	0	0	22	40	0	0	0	0
	8	0	0	0	0	34	26	0	0	23	17	0	0	0	0
R_3	11	0	0	0	0	0	0	21	10	0	0	20	18	1	30
	12	0	0	0	0	0	0	15	30	0	0	20	10	20	5
	13	0	0	0	0	0	0	7	13	0	0	19	20	11	30
	14	0	0	0	0	0	0	15	40	0	0	3	16	21	5

To make our calculations easy to follow, we arrange the substochastic matrix $Q := \{p(x, y) : x, y \in S \setminus D\}$ (dotted in red) which is shown in Table 7. The fundamental matrix N for the matrix Q obtained directly by using formula (4.5) is also given in Table 8.

Table 8: Fundamental matrix $N(F_1(\text{bronze}), F_2(\text{sky blue}), \text{and } F_3(\text{light green}))$.

	T									
R_1	T_1		T_2		T_3		R_2		R_3	
4	1	2	5	6	9	10	7	8	11	12
1.7264	0.8483	1.2045	0.6730	0.8276	0.5930	0.4572	0.4074	0.3871	0.2136	0.1234
0.4534	1.9546	0.8754	0.9603	1.0512	0.9175	0.6871	0.5502	0.5202	0.3247	0.1871
0.5278	0.9825	1.8268	0.9136	1.1530	0.7889	0.6127	0.5601	0.5328	0.2854	0.1651
0.4075	1.2247	1.1191	2.5237	1.6116	1.1843	0.9162	1.1699	1.0814	0.4274	0.2471
0.4196	1.3452	1.0997	1.4805	2.7370	1.0489	0.9196	1.1160	1.0789	0.4094	0.2395
0.3463	1.1765	0.8664	1.0066	1.0697	1.8949	0.7467	0.5690	0.5373	0.4788	0.2585
0.3068	0.8377	0.8954	1.0389	0.9920	0.8523	1.7852	0.5604	0.5267	0.6286	0.3922
0.4117	1.2671	1.1123	2.1569	2.0073	1.1367	0.9174	2.6453	1.8008	0.4211	0.2444
0.4124	1.2742	1.1111	2.0953	2.0738	1.1287	0.9176	1.5619	2.4848	0.4200	0.2440
0.1740	0.5479	0.4621	0.5366	0.5466	0.7778	0.6116	0.2977	0.2805	1.5993	0.4273
0.1986	0.5970	0.5456	0.6333	0.6304	0.7728	0.8554	0.3478	0.3275	0.6447	1.3799

In this example, we will only calculate the row block F_1. The calculation of the other blocks is similar.

In stage 1, we eliminate states $\{4\}$, $\{7, 8\}$, and $\{11, 12\}$ (in this order) to calculate matrix Q^* for the ports T. We also store the probability vectors, $R_s^{(i)}$, for every eliminated state $z_s^{(i)} \in C_i$ for $1 \leq i \leq k$. The matrix Q^* is shown in Table 9.

In stage 2, using formula (4.5), we compute the matrix, N^*, for Q^*, which is given in Table 10.

Table 9: Substochastic matrix Q^*.

1	2	5	6	9	10
0.1135	0.1595	0.1000	0.1000	0.2000	0.1000
0.1716	0.1351	0.0700	0.2000	0.1000	0.0500
0.0700	0.1300	0.2756	0.2344	0.1900	0.1000
0.2000	0.1000	0.2191	0.3309	0.0500	0.1000
0.3100	0.1000	0.1000	0.0800	0.1187	0.1225
0.0300	0.1800	0.1700	0.0500	0.1384	0.1956

Table 10: Fundamental matrix N^*.

1	2	5	6	9	10
1.9546	0.8754	0.9603	1.0512	0.9175	0.6871
0.9825	1.8268	0.9136	1.1530	0.7889	0.6127
1.2247	1.1191	2.5237	1.6116	1.1843	0.9162
1.3452	1.0997	1.4805	2.7370	1.0489	0.9196
1.1765	0.8664	1.0066	1.0697	1.8949	0.7467
0.8377	0.8954	1.0389	0.9920	0.8523	1.7852

Then we use Lemma 4.8 to compute the block $N(1, 1)$ first which is given in Table

11.

Table 11: Insertion of state 4.

4	1	2	5	6	9	10	7	8	11	12
1.7264	0.8483	1.2045	0.6730	0.8276	0.5930	0.4572	0	0	0	0
0.4534	1.9546	0.8754	0.9603	1.0512	0.9175	0.6871	0	0	0	0
0.5278	0.9825	1.8268	0.9136	1.1530	0.7889	0.6127	0	0	0	0

We now calculate the block $N(1,2)$ using Lemma 4.9. For this, we first insert state 7 from $C_2 \subset R_2$. The vector, $R_{7,2}$, in Table 12 represents the probability of transition from states 5 and 6 to the inserted state 7.

Table 12: Insertion of state 7.

p(7,7)	0.3308

	R_7^2
p(5,7)	0.2077
p(6,7)	0.1605

4	1	2	5	6	9	10	7	8	11	12
1.7264	0.8483	1.2045	0.6730	0.8276	0.5930	0.4572	**0.4074**	0	0	0
0.4534	1.9546	0.8754	0.9603	1.0512	0.9175	0.6871	**0.5502**	0	0	0
0.5278	0.9825	1.8268	0.9136	1.1530	0.7889	0.6127	**0.5601**	0	0	0

Similarly, we insert state 8 to obtain Table 13.

Table 13: Insertion of state 8.

p(8,8)	0.17

	R_8^2
p(5,8)	0.1
p(6,8)	0.11
P(7,8)	0.4

4	1	2	5	6	9	10	7	8	11	12
1.7264	0.8483	1.2045	0.6730	0.8276	0.5930	0.4572	0.4074	**0.3871**	0	0
0.4534	1.9546	0.8754	0.9603	1.0512	0.9175	0.6871	0.5502	**0.5202**	0	0
0.5278	0.9825	1.8268	0.9136	1.1530	0.7889	0.6127	0.5601	**0.5328**	0	0

To calculate the block, $N(1,3)$, we insert states 11 and 12 from $C_3 \subset R_3$. In Table 14, we first insert state 11.

Table 14: Insertion of state 11.

p(11,11)	0.24

	R^3_{11}
p(9,11)	0.1067
p(10,11)	0.2167

4	1	2	5	6	9	10	7	8	11	12
1.7264	0.8483	1.2045	0.6730	0.8276	0.5930	0.4572	0.4074	0.3871	**0.2136**	0
0.4534	1.9546	0.8754	0.9603	1.0512	0.9175	0.6871	0.5502	0.5202	**0.3247**	0
0.5278	0.9825	1.8268	0.9136	1.1530	0.7889	0.6127	0.5601	0.5328	**0.2854**	0

Finally, we insert state 12 in Table 15 and aggregating $N(1,1), N(1,2)$ and $N(1,3)$, we have calculated the block F_1.

Table 15: Insertion of state 12.

p(12,12)	0.10

	R^3_{12}
p(9,12)	0.03
p(10,12)	0.12
p(11,12)	0.18

4	1	2	5	6	9	10	7	8	11	12
1.7264	0.8483	1.2045	0.6730	0.8276	0.5930	0.4572	0.4074	0.3871	0.2136	**0.1234**
0.4534	1.9546	0.8754	0.9603	1.0512	0.9175	0.6871	0.5502	0.5202	0.3247	**0.1871**
0.5278	0.9825	1.8268	0.9136	1.1530	0.7889	0.6127	0.5601	0.5328	0.2854	**0.1651**

CHAPTER 5: CONCLUSION

If our Markov model follows the state space configuration (or clustering) of the IP model, then the algorithms we presented in this thesis have an advantage over other direct and iterative methods. Because our algorithms give exact results, the accuracy of our results are superior to any other iterative methods. For any IP Markov model, large or small, it makes no difference whether we test our algorithm on a large model or a small model. In this case, the numerical examples given in this thesis are sufficient. The key issue is actually identifying the number of clusters within the state space itself. Finding the number of clusters, in any field of research, is not a trivial problem. It is well-known that such problems are classified as NP hard. We consider two possible cases where our approach can be useful.

Let us assume that we know the exact number of clusters in our matrix but they are not ordered cleanly in blocks (as we assumed in Chapter 3). In this case, because the number of clusters is known a priori, it is simply a permutation problem to identify which states belong to which clusters. We claim that one can simply use K-means clustering algorithm to group states into clusters.

The second case deals with the possibility of using the IP model as an approximation of the nearly decomposable structure that can be found in transition matrices of many Markov models , for example, in the Web. We can use the approximation method discussed in Chapter 3 section 12 where we can use ϵ_{max} as a tuning param-

eter to transform a block structure with many small transitions to an IP model by redistributing small probabilities back to islands. Although the resulting IP model is an approximation of the original model, we can take advantage of the possibility of parallel computing that is feasible for the IP model and that our algorithms allow for. In this case, we lose the precision in our results but we still benefit from parallel computing. Thus, the results in this thesis can be extended to approximate transition matrices of nearly decoupled Markov models. For this, we can begin by studying the effect of the fine turning parameter ϵ_{max}.

REFERENCES

[1] K. Avrachenkov, N. Litvak, and K. S. Pham. A singular perturbation approach for choosing the PageRank damping factor. *Internet Mathematics*, 5(1-2):47–69, 2008.

[2] K. Avrachenkov, D. Nemirovsky, and K. S. Pham. A survey on distributed approaches to graph based reputation measures. In *The proceedings of the 2nd international conference on Performance evaluation methodologies and tools*, page 82. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[3] K. E. Avrachenkov, J. A. Filar, and P. G. Howlett. *Analytic perturbation theory and its applications*, volume 135. SIAM, 2013.

[4] G. E. Cho and C. D. Meyer. Markov chain sensitivity measured by mean first passage times. *Linear Algebra and its Applications*, 316(1-3):21–28, 2000.

[5] G. E. Cho and C. D. Meyer. Comparison of perturbation bounds for the stationary distribution of a markov chain. *Linear Algebra and its Applications*, 335(1-3):137–150, 2001.

[6] T. Dayar and W. J. Stewart. On the effects of using the Grassmann-Taksar-Heyman method in iterative aggregation-disaggregation. *SIAM Journal on Scientific Computing*, 17(1):287–303, 1996.

[7] A. Gambin and P. Pokarowski. A new combinatorial algorithm for large Markov chains. In *Computer Algebra in Scientific Computing CASC 2001*, pages 195–211. Springer, 2001.

[8] W. K. Grassmann and D. A. Stanford. Matrix analytic methods. In *Computational probability*, pages 153–203. Springer, 2000.

[9] W. K. Grassmann, M. I. Taksar, and D. P. Heyman. Regenerative analysis and steady state distributions for Markov chains. *Operations Research*, 33(5):1107–1116, 1985.

[10] R. Hassin and M. Haviv. Mean passage times and nearly uncoupled markov chains. *SIAM Journal on Discrete Mathematics*, 5(3):386–397, 1992.

[11] M. Haviv and L. Van der Heyden. Perturbation bounds for the stationary probabilities of a finite markov chain. *Advances in Applied Probability*, 16(04):804–818, 1984.

[12] D. P. Heyman. Accurate computation of the fundamental matrix of a markov chain. *SIAM journal on matrix analysis and applications*, 16(3):954–963, 1995.

[13] J. J. Hunter. Accurate calculations of stationary distributions and mean first passage times in markov renewal processes and markov chains. *Special Matrices*, 4(1):151–175, 2016.

[14] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub. Exploiting the block structure of the web for computing PageRank. *Stanford University Technical Report*, 2003.

[15] J. G. Kemeny and J. L. Snell. *Finite Markov chains: With a New Appendix "Generalization of a Fundamental Matrix"*. Springer-Verlag New York, 1976.

[16] J. Koury, D. McAllister, and W. J. Stewart. Iterative methods for computing stationary distributions of nearly completely decomposable Markov chains. *SIAM Journal on Algebraic Discrete Methods*, 5(2):164–186, 1984.

[17] A. N. Langville and C. D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, 2011.

[18] N. Liu and W. J. Stewart. Markov chains and spectral clustering. In *Performance Evaluation of Computer and Communication Systems. Milestones and Future Challenges*, pages 87–98. Springer, 2011.

[19] C. D. Meyer. Stochastic complementation, uncoupling Markov chains, and the theory of nearly reducible systems. *SIAM Review*, 31(2):240–272, 1989.

[20] C. D. Meyer and C. D. Wessell. Stochastic data clustering. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1214–1236, 2012.

[21] C. A. O'Cinneide. Entrywise perturbation theory and error analysis for Markov chains. *Numerische Mathematik*, 65(1):109–120, 1993.

[22] K. Reichel, V. Bahier, C. Midoux, N. Parisey, J.-P. Masson, and S. Stoeckel. Interpretation and approximation tools for big, dense Markov chain transition matrices in population genetics. *Algorithms for Molecular Biology*, 10(1):1, 2015.

[23] Y. Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[24] P. J. Schweitzer. Perturbation theory and finite markov chains. *Journal of Applied Probability*, 5(02):401–413, 1968.

[25] T. J. Sheskin. A Markov chain partitioning algorithm for computing steady state probabilities. *Operations Research*, 33(1):228–235, 1985.

[26] I. Sonin. The elimination algorithm for the problem of optimal stopping. *Mathematical Methods of Operations Research*, 49(1):111–123, 1999.

[27] I. Sonin. The state reduction and related algorithms and their applications to the study of Markov chains, graph theory, and the optimal stopping problem. *Advances in Mathematics*, 145(2):159–188, 1999.

[28] I. Sonin and C. Steinberg. Continue, quit, restart probability model. *Annals of Operations Research*, 241(1-2):295–318, 2012,2016.

[29] I. Sonin and C. Steinberg. Elimination and insertion operations for finite Markov chains. In *Modern Trends in Controlled Stochastic Processes: Theory and Applications* . The University of Liverpool, 2015.

[30] I. Sonin and J. Thornton. Recursive algorithm for the fundamental/group inverse matrix of a Markov chain from an explicit formula. *SIAM Journal on Matrix Analysis and Applications*, 23(1):209–224, 2001.

[31] I. M. Sonin. The optimal stopping of a Markov chain and recursive solution of Poisson and Bellman equations. In *From Stochastic Calculus to Mathematical Finance: The Shiryaev Festschrift*, pages 609–621. Springer Berlin Heidelberg, 2006.

[32] I. M. Sonin. A generalized Gittins index for a Markov chain and its recursive calculation. *Statistics & Probability Letters*, 78(12):1526–1533, 2008.

[33] W. J. Stewart. *Probability, Markov Chains, Queues, and Simulation: The Mathematical Basis of Performance Modeling*. Princeton University Press, 2009.

[34] Y. Q. Zhao. Censoring technique in studying block-structured markov chains. *Advances in Algorithmic Methods for Stochastic Models*, 417:433, 2000.

APPENDIX

Proofs

Proof of Lemma 2.5: We need to show that $\tilde{\pi}_C$ is the invariant distribution of P_D. Let $\pi = (\pi_D, \pi_C)$ be the stationary distribution for D and C respectively. From the relation (2.15), we have

$$[\pi_D \;\; \pi_C] \begin{bmatrix} Q & T \\ R & K \end{bmatrix} = [\pi_D \;\; \pi_C], \tag{5.1}$$

Therefore, we have

$$\pi_D = \pi_D Q + \pi_C R, \tag{5.2}$$

and

$$\pi_C = \pi_D T + \pi_C K. \tag{5.3}$$

Solving these system of equations, we obtain

$$\pi_D = \pi_C R(I - Q)^{-1} = \pi_C R N_D. \tag{5.4}$$

Substituting equation (5.4) in (5.3), we obtain

$$\pi_C = \pi_C R N_D T + \pi_C K = \pi_C (K + R N_D T) = \pi_C P_D, \tag{5.5}$$

where the last equality follows from Lemma 2.1. Normalizing π_C by $\sum_{i\in C}\pi_i$, we obtain $\bar{\pi}_C = \bar{\pi}_C P_D$. Since the invariant distributions of ergodic MCs are unique, π_C is the invariant distribution of P_D. □

Proof of Lemma 2.4: If we apply formula (2.8) for $x = z$, we obtain the equality

$$w_{k+1}(z,y) = \frac{w_k(z,y)}{(1-w_k(z,z))}, \tag{5.6}$$

$$w_{k+1}(z,y)(1-w_k(z,z)) = w_k(z,y). \tag{5.7}$$

We replace $w_{k+1}(z,y)$ given by (5.6) in formula (2.8) and solve for $w_k(\cdot,y)$ to obtain

$$w_k(\cdot,y) = w_{k+1}(\cdot,y) - w_k(\cdot,z)w_{k+1}(z,y), \quad y \in S. \tag{5.8}$$

Now we applying formula (5.8) for $y = z$ gives

$$w_k(\cdot,z) = \frac{w_{k+1}(\cdot,z)}{(1+w_{k+1}(z,z))}. \tag{5.9}$$

By substituting formula (5.9) for $w_k(\cdot,z)$ in (2.8), we obtain insertion formula (2.12).

□

Proof of Lemma 3.1: (a) Since $\pi(S_1) = 1$ and $\pi(S_1) = \sum_{y\in S_1}\pi_1(y) = \pi_1(z) + \sum_{y\in S_2}\pi_1(y)$. Let $\alpha_1 = \sum_{y\in S_2}\pi_1(y)$, then $\pi_1(z) = 1-\alpha_1$. The equality $\pi_1(y) = \alpha_1\pi_2(y), y \in S_2$ follows from Lemma 2.5. To prove formula (3.3), we consider the balance equation at z,

$$\begin{aligned}\pi_1(z) &= \sum_{y\in S_1}\pi_1(y)p_1(y,z),\\ &= \sum_{y\in S_2}\pi_1(y)p_1(y,z) + \pi_1(z)p_1(z,z), \qquad y \neq z,\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{1-p_1(z,z)}\Big(\sum_{y\in S_2}\pi_1(y)p_1(y,z)\Big),\\
&\overset{(by\ 3.2)}{=} \alpha_1\frac{1}{s_1(z)}\Big(\sum_{y\in S_2}\pi_2(y)p_1(y,z)\Big), \qquad \text{where } s_1(z)=1-p_1(z,z),\\
&= \alpha_1\frac{R_1(z)}{s_1(z)},\\
&\overset{(by\ 3.2)}{=} (1-\pi_1(z))\,\frac{R_1}{s_1(z)},\\
&= \frac{R_1(z)}{R_1(z)+s_1(z)}, \qquad (5.10)\\
&= \beta_1 R_1.
\end{aligned}
$$

Identity for α_1 can easily be derived from (3.2),

$$\alpha_1 = 1-\frac{s_1(z)}{R_1(z)} \overset{(by\ 5.10)}{=} \frac{s_1}{R_1+s_1}.$$

□

Proof of Lemma 4.1: The proof for equality (a) follows from Definition 1 by conditioning on an initial transition to some intermediate state $z \in D, z \neq y$, and counting the first visit as 1 if $x = y$,

$$
\begin{aligned}
n(x,y) &= E\Big[\sum_{n=0}^{\tau-1}\mathbb{1}_{\{Z_n=y\}}|Z_0=x\Big], \qquad (5.11)\\
&= \mathbb{1}_x(y)+\sum_{n=0}^{\tau-1}\sum_{z\in D}P(Z_n=y|Z_1=z,Z_0=x)p(x,z),\\
&= \mathbb{1}_x(y)+\sum_{z\in D}p(x,z)E\Big[\sum_{n=0}^{\tau-1}\mathbb{1}_{\{Z_n=y\}}|Z_1=z\Big],\\
&= \mathbb{1}_x(y)+\sum_{z\in D}p(x,z)n(z,y).
\end{aligned}
$$

In matrix form the last equation can be written as $N = I + QN$.

Equality (b) can be proved, in a similar way, by conditioning on the time of the last exit to y,

$$\begin{aligned}
n(x,y) &= E\Big[\sum_{n=0}^{\tau-1} \mathbb{1}_{\{Z_n=y\}} | Z_0 = x\Big], \\
&= \mathbb{1}_x(y) + \sum_{n=0}^{\tau-1} \sum_{z\in D} P(Z_n = y | Z_{n-1} = z, Z_0 = x) P(Z_{n-1} = z | Z_0 = x), \\
&= \mathbb{1}_x(y) + \sum_{z\in D} p(z,y) \sum_{n=0}^{\tau-1} P(Z_{n-1} = z | Z_0 = x), \\
&= \mathbb{1}_x(y) + \sum_{z\in D} p(z,y) E\Big[\sum_{n=0}^{\tau-1} \mathbb{1}_{\{Z_{n-1}=z\}} | Z_0 = x\Big], \\
&= \mathbb{1}_x(y) + \sum_{z\in D} n(x,z) p(z,y).
\end{aligned}$$

In matrix form, the last equation can be written as $N = I + NQ$. □

Remark 5.1. It is also possible to prove the equalities in (4.4) algebraically. Starting with (4.3) $N = I + Q + Q^2 + \cdots$ and left multiplication by Q gives $QN = Q(I + Q + Q^2 + \cdots) = Q + Q^2 + Q^3 + \cdots = (I + Q + Q^2 + \cdots)Q = NQ$.

Proof of Lemma 4.2: For any state $x \in D$ and $y \in C$, MC (Z_n) can either transition to C in one step, in which case $\tau = 1$, or can transition to state $z \in D$, so,

$$\begin{aligned}
u(x,y) &= P(Z_\tau = y | Z_0 = x), \\
&= P(Z_\tau = y, \tau = 1 | Z_0 = x) + P(Z_\tau = y, \tau > 1 | Z_0 = x), \\
&= p(x,y) + \sum_{z\in D} p(Z_\tau = y, \tau > 1 | Z_1 = z) p(x,z), \\
&= p(x,y) + \sum_{z\in D} u(z,y) p(x,z).
\end{aligned}$$

In matrix form, the last equation is $U = T + QU = (I - Q)^{-1}T \overset{(by\ 4.5)}{=} NT$.

□

Proof of Proposition 4.3: Let $x, y \in S_2 \setminus G$. And let

$$V_{(x,y)}^{(i)} = \sum_{n=0}^{\tau^{(i)}-1} \mathbb{1}_{\{Z_n=y|Z_n=x\}}$$

be the number of visits of MC $(Z_n^i), i = 1, 2$, to state y from state x. Notice that the elimination of set D only reduces the length visits to y but number of visits $V_{(x,y)}^{(1)} = V_{(x,y)}^{(2)}$ for MCs (Z_n^1) and (Z_n^2), respectively. Let $\tau^D = \tau^{(1)} - \tau^{(2)}$. We now need to show $n_1(x, y) = n_2(x, y)$. By definition 1, expectation under the probability distribution P_1 gives

$$\begin{aligned} n^1(x,y) &= E^1\Big[\sum_{n=0}^{\tau^{(1)}-1} \mathbb{1}_{\{Z_n^1=y\}}|Z_0 = x\Big], \\ &= \sum_{n=0}^{\tau^{(2)}-1} P_1(Z_n^1 = y|Z_0 = x) + \sum_{z\in D} P_1(Z_{\tau^D}^1 = y|Z_1 = z)p_1(x,z), \\ &= \sum_{n=0}^{\tau^{(2)}-1} P_2(Z_n^2 = y|Z_0 = x) = E^2\Big[\sum_{n=0}^{\tau^{(2)}-1} \mathbb{1}_{\{Z_n^2=y\}}|Z_0 = x\Big] = n^2(x,y), \end{aligned}$$

where the second last equality follows because the probability distribution $P_2 = K + RU$ as defined in Lemma 2.1 and $V_{(x,y)}^{(1)} = V_{(x,y)}^{(2)}$. Proof of $u_1^G(x, y) = u_2^G(x, y)$ follows from Lemma 4.2 for all $x \in S_1 \setminus G$ and $y \in G$ since $p_1(x, y) = p_2(x, y)$. □

Algorithms

Algorithm 1 The GTH/S Algorithm

procedure (FORWARD STAGE)
  for $i = r, r-1 ... 2$ **do** ▷ Assume P is $r \times r$ matrix
    $P_2 \leftarrow \dots P_i \cdots \leftarrow P_{r-1} \leftarrow P_r$ ▷ $|P_2| = 2$ (two states)
  end for
end procedure
procedure (BACKWARD STAGE)
  $a_2 = (1 - p_{22})^{-1} p_{12} a_1 \leftarrow a^T P_{r-1} = a^T$ ▷ $a^T = [a_1\ a_2]$
  $k_2 \leftarrow (1 - p_{22})^{-1} p_{22}$
  for $j = 3, 4 .....r$ **do**
    Compute $k_j = (1 - p_{jj})^{-1}(p_{1j} + \sum_{i=2}^{r-1} p_{ij} k_i)$
  end for
  Compute the normalization condition $a_1 = (1 + \sum_{i=2}^{r} k_i)$
  for i=1,2.....r **do**
    $a_i = k_i a_1$ ▷ a is the stationary vector
  end for
end procedure

Algorithm 2 The IP Algorithm

procedure (STAGE 1)
  for $i, j = 1, 2...k, i \neq j$ **do**
    $P^*_{ii} \leftarrow P_i$ ▷ eliminate set R_i from L_i
    $P_{ij} \leftarrow P$ ▷ Obtain (unchanged) blocks P_{ij}
  end for
  Obtain matrix P^* ▷ Obtain model $M^* = (T, P^*)$
end procedure
procedure (STAGE 2)
  $\pi^* \leftarrow P^*$ ▷ invariant distribution for model M^*
end procedure
procedure (STAGE 3)(step 1)
  for $i = 1, 2...k$, **do**
    $\pi_i \leftarrow P^*_i$ ▷ invariant distribution for model $M_i = (L^*_i, P^*_i)$
    $\lambda_i \leftarrow w_i \leftarrow \pi_i$ ▷ coefficients for model M_i
  end for
end procedure
procedure (STAGE 3)(step 2)
  for $i = 1, 2...k$ **do**
    $e_i \leftarrow \lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ ▷ $e_i = \pi(L^*_i)$
    $\pi \leftarrow e_i$ ▷ invariant distribution for model $M = (S, P)$
  end for
end procedure

Algorithm 3 The FUNDQ Algorithm

Let $S_1 = \{1, 2, \ldots, k-1, k, \ldots, n\}$.
procedure (STAGE 1. FORWARD STAGE)
  $W_k \leftarrow \ldots W_2 \leftarrow W_1 = Q_1$ ▷ (eliminate states $(1, 2, \ldots, k-1)$)
  $N_k \leftarrow Q_k \leftarrow W_k$ ▷ $N_k = (I - Q_k)^{-1}$
end procedure
procedure (STAGE 2. BACKWARD STAGE)
  $N_1 \leftarrow N_2 \leftarrow \ldots N_{k-1} \leftarrow N_k$ ▷ (insert states $(1, 2, \ldots, k-1)$)
end procedure

Algorithm 4 The IP FUND Algorithm

1: **procedure** (STAGE 1)
2: **for** $i = 1, 2, ...k$ **do**
3: $Q^*_{ii} \leftarrow Q_i$, store $\{\mathbf{R}_1^{(i)}, \mathbf{R}_2^{(i)}, \ldots, \mathbf{R}_{c_i}^{(i)}\}$
4: Obtain matrix Q^* ▷ Q^* is substochastic matrix for T
5: **end for**
6: **end procedure**
7: **procedure** (STAGE 2)
8: $N^* \leftarrow Q^*$ ▷ $N^* = (I - Q^*)^{-1}$
9: **end procedure**
10: **procedure** (STAGE 3)
11: **for** $i = 1, 2, ...k$ **do**
12: $N(i,i) \leftarrow N_i \leftarrow N^*$ ▷ $N(i,i) \subset N_i$
13: **for** $j = 1, 2, \ldots, k, j \neq i$, **do**
14: $N(i,j) \leftarrow \mathbf{m}_1 \leftarrow \mathbf{m}_2 \cdots \leftarrow \mathbf{m}_{c_i}. \leftarrow N(i,i)$
15: **end for**
16: Compute F_i ▷ F_i is a row block of matrix N for island L_i
17: **end for**
18: Obtain matrix N
19: **end procedure**