Programming Biological Time Machines: Lecture II[1][2]

Burak Kocuk

Sabancı University

joint work with Ali Rana Atılgan and Oğuz Mesüm

17 November 2023

[1]This research is funded by The Scientific and Technological Research Council of Turkey with project number 120C151.

[2]Based on: Mesum, O., Atilgan, A. R., & Kocuk, B. (2023). A Stochastic Programming Approach to the Antibiotics Time Machine Problem. bioRxiv.

Outline

1. Introduction
2. Deterministic Model
3. Stochastic Model
4. Computational Results
5. Conclusion

Outline

1. Introduction
   - Roadmap
2. Deterministic Model
3. Stochastic Model
4. Computational Results
5. Conclusion

Roadmap

In this presentation, I will

- review the MILP formulation of the *Static Deterministic* Version,

Roadmap

In this presentation, I will

- review the MILP formulation of the *Static Deterministic* Version,
- propose a dynamic programming (DP) formulation of the *Dynamic Deterministic* Version, in which an antibiotic treatment policy is sought,

Roadmap

In this presentation, I will

- review the MILP formulation of the *Static Deterministic* Version,
- propose a dynamic programming (DP) formulation of the *Dynamic Deterministic* Version, in which an antibiotic treatment policy is sought,
- propose an MILP formulation of the *Static Stochastic* Version, in which the antibiotic matrices are uncertain,

Roadmap

In this presentation, I will

- review the MILP formulation of the *Static Deterministic* Version,
- propose a dynamic programming (DP) formulation of the *Dynamic Deterministic* Version, in which an antibiotic treatment policy is sought,
- propose an MILP formulation of the *Static Stochastic* Version, in which the antibiotic matrices are uncertain,
- propose a DP formulation of the *Dynamic Stochastic* Version, in which an antibiotic treatment policy is sought and the antibiotic matrices are uncertain.

Roadmap

In this presentation, I will

- review the MILP formulation of the *Static Deterministic* Version,
- propose a dynamic programming (DP) formulation of the *Dynamic Deterministic* Version, in which an antibiotic treatment policy is sought,
- propose an MILP formulation of the *Static Stochastic* Version, in which the antibiotic matrices are uncertain,
- propose a DP formulation of the *Dynamic Stochastic* Version, in which an antibiotic treatment policy is sought and the antibiotic matrices are uncertain.

I will also show how to adapt a Sample Average Approximation approach that exploits the special structure of the problem and provide accurate solutions that perform very well in an out-of-sample analysis for the stochastic versions.

Outline

1 Introduction

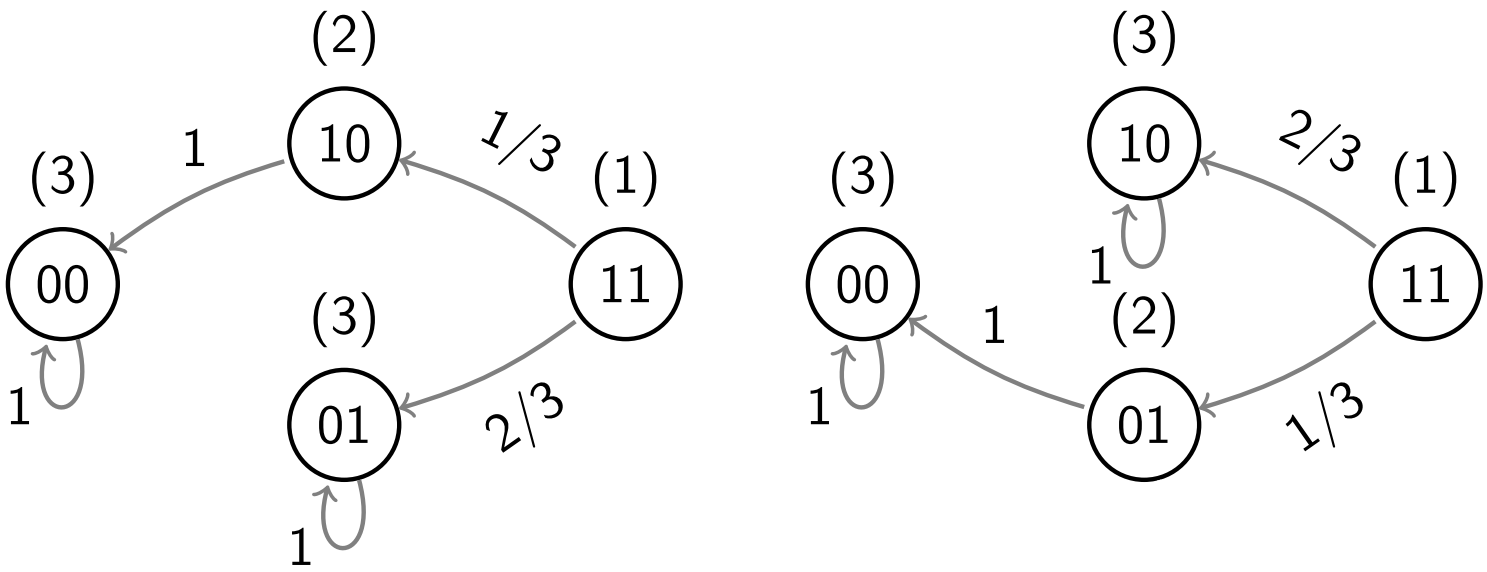
2 Deterministic Model
- Problem Definition
- Static Optimization
- Dynamic Optimization

3 Stochastic Model

4 Computational Results

5 Conclusion

An Example with $a = 2$, $K = 2$ and $N = 2$ (initial $= 11$)

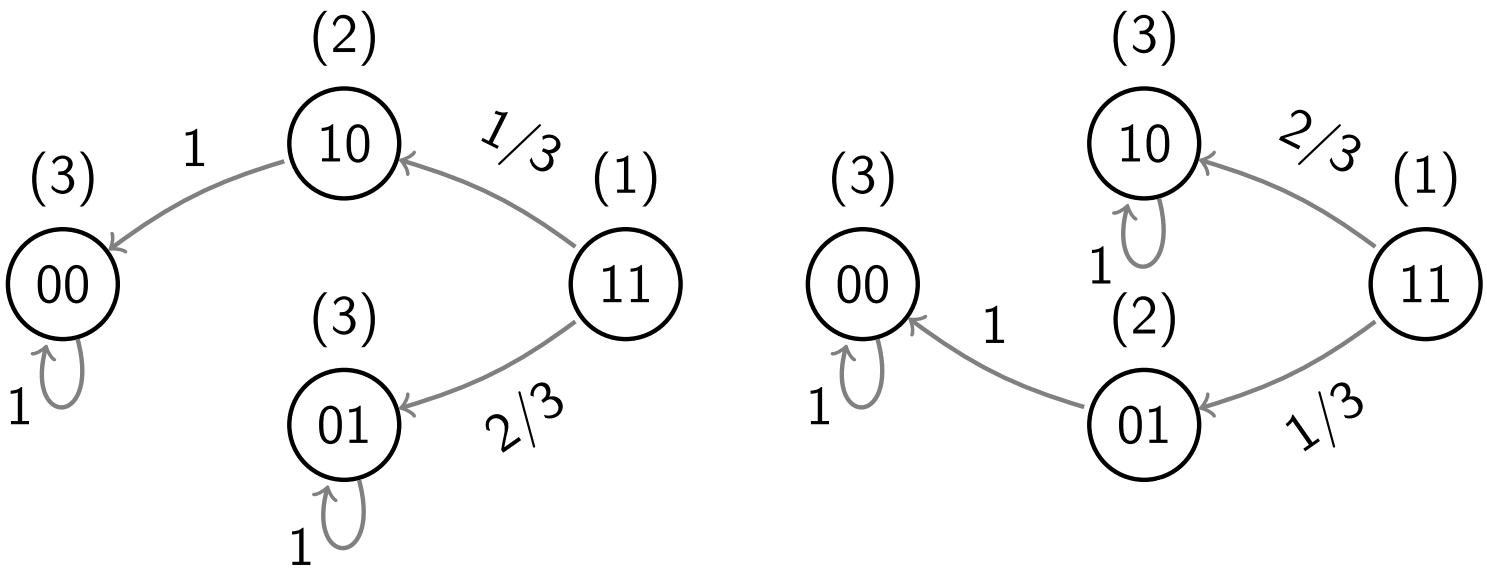


(a) Antibiotic I.

(b) Antibiotic II.

Figure: Markov chains corresponding to two antibiotics.

An Example with $a = 2$, $K = 2$ and $N = 2$ (initial = 11)



(a) Antibiotic I.

(b) Antibiotic II.

Figure: Markov chains corresponding to two antibiotics.

Static: 4 possible sequences
Optimal Value: 2/3
Optimal Sequence: I-II or II-I

Dynamic: 8 possible policies
Optimal Value: 1
Optimal Policy: I-(I,II) or II-(I,II)

Problem Definitions

Problem (Static Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Problem Definitions

Problem (Static Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Problem (Dynamic Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *policy* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Static Optimization

Problem (Static Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Static Optimization

Problem (Static Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Nonlinear Formulation:

$$\max \boldsymbol{p}^T \left(\prod_{n=1}^{N} \boldsymbol{T_n} \right) \boldsymbol{q}$$

$$\text{s.t. } \boldsymbol{T_n} \in \mathcal{M} := \{\boldsymbol{M^k} : k = 1, \ldots, K\} \qquad n = 1, \ldots, N.$$

Static Optimization

Problem (Static Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Nonlinear Formulation:

$$\max \boldsymbol{p}^T \left(\prod_{n=1}^{N} \boldsymbol{T_n} \right) \boldsymbol{q}$$

$$\text{s.t. } \boldsymbol{T_n} \in \mathcal{M} := \{\boldsymbol{M^k} : k = 1, \ldots, K\} \qquad n = 1, \ldots, N.$$

Recall that there exists an MILP Formulation for this version.

Static Optimization: MILP Formulation

- The key idea is to define two sets of decisions variables:
  - $x_{n,k}$: one if antibiotic k is applied in step n and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}_+^d$: the probability distribution after n transitions.

Static Optimization: MILP Formulation

- The key idea is to define two sets of decisions variables:
  - $x_{n,k}$: one if antibiotic k is applied in step n and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}_+^d$: the probability distribution after n transitions.

- To ensure exactly one antibiotic selection per step, we must have

$$\sum_{k=1}^{K} x_{n,k} = 1 \qquad \forall n.$$

Static Optimization: MILP Formulation

- The key idea is to define two sets of decisions variables:
  - $x_{n,k}$: one if antibiotic k is applied in step n and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}_+^d$: the probability distribution after n transitions.

- To ensure exactly one antibiotic selection per step, we must have

$$\sum_{k=1}^{K} x_{n,k} = 1 \qquad \forall n.$$

- Notice that these variables satisfy the recursion

$$\boldsymbol{u_n}^\top = \sum_{k=1}^{K} \boldsymbol{u_{n-1}}^\top \boldsymbol{M^k} x_{n,k} \qquad \forall n$$

where $\boldsymbol{u_0} = \boldsymbol{p}$.

Static Optimization: MILP Formulation

- The key idea is to define two sets of decisions variables:
  - $x_{n,k}$: one if antibiotic k is applied in step n and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}_+^d$: the probability distribution after n transitions.

- To ensure exactly one antibiotic selection per step, we must have

$$\sum_{k=1}^{K} x_{n,k} = 1 \qquad \forall n.$$

- Notice that these variables satisfy the recursion

$$\boldsymbol{u_n}^\top = \sum_{k=1}^{K} \boldsymbol{u_{n-1}}^\top \boldsymbol{M^k} x_{n,k} \qquad \forall n$$

  where $\boldsymbol{u_0} = \boldsymbol{p}$.

- This relation is linearized using a disjunctive formulation by defining copy variables $\boldsymbol{v_{n,k}} \in \mathbb{R}_+^d$, which take value $\boldsymbol{u_{n-1}}$ if $x_{n,k} = 1$.

Dynamic Optimization

Problem (Dynamic Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *policy* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Dynamic Optimization

Problem (Dynamic Deterministic Version)

Given K antibiotics with their transition probability matrices $\boldsymbol{M^k}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *policy* of antibiotics to be applied so that the probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

The key difference between this version and its static counterpart is that the decision at step n can be a function of the state we are in since we assume that the state transitions are observable.

Dynamic Optimization: MILP Formulation

- We change the definition of the binary variables to account for the flexibility.
  - $x_{n,k,i}$: one if antibiotic k is applied in step n *when the state is* i and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}_+^d$: the probability distribution after n transitions.

Dynamic Optimization: MILP Formulation

- We change the definition of the binary variables to account for the flexibility.
  - $x_{n,k,i}$: one if antibiotic k is applied in step n *when the state is* i and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}_+^d$: the probability distribution after n transitions.
- To ensure exactly one antibiotic selection per step at each state:

$$\sum_{k=1}^{K} x_{n,k,i} = 1 \qquad \forall n, i.$$

Dynamic Optimization: MILP Formulation

- We change the definition of the binary variables to account for the flexibility.
  - $x_{n,k,i}$: one if antibiotic k is applied in step n *when the state is* i and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}^d_+$: the probability distribution after n transitions.
- To ensure exactly one antibiotic selection per step at each state:

$$\sum_{k=1}^{K} x_{n,k,i} = 1 \qquad \forall n, i.$$

- Notice that these variables satisfy the recursion

$$u_{n,j} = \sum_{k=1}^{K} \sum_{i=1}^{d} u_{n-1,i} M^k_{ij} x_{n,k,i} \qquad n = 1, \ldots, N, j = 1, \ldots, d$$

Dynamic Optimization: MILP Formulation

- We change the definition of the binary variables to account for the flexibility.
  - $x_{n,k,i}$: one if antibiotic k is applied in step n *when the state is* i and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}^d_+$: the probability distribution after n transitions.
- To ensure exactly one antibiotic selection per step at each state:

$$\sum_{k=1}^{K} x_{n,k,i} = 1 \qquad \forall n, i.$$

- Notice that these variables satisfy the recursion

$$u_{n,j} = \sum_{k=1}^{K} \sum_{i=1}^{d} u_{n-1,i} M^k_{ij} x_{n,k,i} \qquad n = 1, \ldots, N, j = 1, \ldots, d$$

- Unfortunately, disjunction idea does not extend to this case.

Dynamic Optimization: MILP Formulation

- We change the definition of the binary variables to account for the flexibility.
  - $x_{n,k,i}$: one if antibiotic k is applied in step n *when the state is* i and zero otherwise
  - $\boldsymbol{u_n} \in \mathbb{R}^d_+$: the probability distribution after n transitions.
- To ensure exactly one antibiotic selection per step at each state:

$$\sum_{k=1}^{K} x_{n,k,i} = 1 \qquad \forall n, i.$$

- Notice that these variables satisfy the recursion

$$u_{n,j} = \sum_{k=1}^{K} \sum_{i=1}^{d} u_{n-1,i} M^k_{ij} x_{n,k,i} \qquad n = 1, \ldots, N, j = 1, \ldots, d$$

- Unfortunately, disjunction idea does not extend to this case.
- Instead, we define auxiliary variables $w_{n-1,k,i} := u_{n-1,i} x_{n,k,i}$ and linearize this relation using the McCormick envelops.

Dynamic Optimization: MILP Formulation

$$\max_{\boldsymbol{u},\boldsymbol{w},\boldsymbol{x}} \ \boldsymbol{q}^\top \boldsymbol{u_N} \qquad \text{(maximize probability)}$$

$$\text{s.t. } \boldsymbol{u_0} = \boldsymbol{p} \qquad \text{(initial condition)}$$

$$u_{n,j} = \sum_{k=1}^{K} \sum_{i=1}^{d} M_{ij}^{k} w_{n-1,k,i} \qquad \forall n, j \qquad \text{(linearization)}$$

$$w_{n-1,k,i} \geq 0 \qquad \forall n, k, i \qquad \text{(McCormick-1)}$$

$$w_{n-1,k,i} \geq u_{n-1,i} + x_{n,k,i} - 1 \qquad \forall n, k, i \qquad \text{(McCormick-2)}$$

$$w_{n-1,k,i} \leq u_{n-1,i}, \ \ w_{n-1,k,i} \leq x_{n,k,i} \qquad \forall n, k, i \qquad \text{(McCormick-3,4)}$$

$$\sum_{k=1}^{K} x_{n,k,i} = 1 \qquad \forall n, i \qquad \text{(one drug/step,state)}$$

$$\sum_{j=1}^{d} u_{n,j} = 1 \qquad \forall n \qquad \text{(valid equality)}$$

$$\boldsymbol{u_n} \in \mathbb{R}_{+}^{d}, \ \ x_{n,k,i} \in \{0,1\} \qquad \forall n, k, i \qquad \text{(domains)}$$

Dynamic Optimization: MILP Formulation

$$\max_{\boldsymbol{u},\boldsymbol{w},\boldsymbol{x}} \ \boldsymbol{q}^\top \boldsymbol{u_N} \qquad \text{(maximize probability)}$$

$$\text{s.t. } \boldsymbol{u_0} = \boldsymbol{p} \qquad \text{(initial condition)}$$

$$u_{n,j} = \sum_{k=1}^{K} \sum_{i=1}^{d} M_{ij}^{k} w_{n-1,k,i} \qquad \forall n, j \qquad \text{(linearization)}$$

$$w_{n-1,k,i} \geq 0 \qquad \forall n, k, i \qquad \text{(McCormick-1)}$$

$$w_{n-1,k,i} \geq u_{n-1,i} + x_{n,k,i} - 1 \qquad \forall n, k, i \qquad \text{(McCormick-2)}$$

$$w_{n-1,k,i} \leq u_{n-1,i}, \ \ w_{n-1,k,i} \leq x_{n,k,i} \qquad \forall n, k, i \qquad \text{(McCormick-3,4)}$$

$$\sum_{k=1}^{K} x_{n,k,i} = 1 \qquad \forall n, i \qquad \text{(one drug/step,state)}$$

$$\sum_{j=1}^{d} u_{n,j} = 1 \qquad \forall n \qquad \text{(valid equality)}$$

$$\boldsymbol{u_n} \in \mathbb{R}_{+}^{d}, \ \ x_{n,k,i} \in \{0,1\} \qquad \forall n, k, i \qquad \text{(domains)}$$

Unfortunately, this MILP does not scale well computationally.

Dynamic Optimization: DP Formulation

- Assume that the initial and final distribution vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ are unit vectors corresponding to states i_I and i_F, respectively.

Dynamic Optimization: DP Formulation

- Assume that the initial and final distribution vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ are unit vectors corresponding to states i_I and i_F, respectively.
- Let us define $\nu_n(i)$ as the maximum probability of reaching the final state i_F at the end of N-th transition given that we are at state i after n transitions.

Dynamic Optimization: DP Formulation

- Assume that the initial and final distribution vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ are unit vectors corresponding to states i_I and i_F, respectively.
- Let us define $\nu_n(i)$ as the maximum probability of reaching the final state i_F at the end of N-th transition given that we are at state i after n transitions.
- With this definition, we have the following boundary value:

$$\nu_N(i) = \begin{cases} 1 & \text{if } i = i_F \\ 0 & \text{if } i \neq i_F \end{cases}.$$

Dynamic Optimization: DP Formulation

- Assume that the initial and final distribution vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ are unit vectors corresponding to states i_I and i_F, respectively.
- Let us define $\nu_n(i)$ as the maximum probability of reaching the final state i_F at the end of N-th transition given that we are at state i after n transitions.
- With this definition, we have the following boundary value:

$$\nu_N(i) = \begin{cases} 1 & \text{if } i = i_F \\ 0 & \text{if } i \neq i_F \end{cases}.$$

- Solving the dynamic version of the antibiotics time machine problem amounts to computing $\nu_0(i_I)$.

Dynamic Optimization: DP Formulation

- Assume that the initial and final distribution vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ are unit vectors corresponding to states i_I and i_F, respectively.
- Let us define $\nu_n(i)$ as the maximum probability of reaching the final state i_F at the end of N-th transition given that we are at state i after n transitions.
- With this definition, we have the following boundary value:

$$\nu_N(i) = \begin{cases} 1 & \text{if } i = i_F \\ 0 & \text{if } i \neq i_F \end{cases}.$$

- Solving the dynamic version of the antibiotics time machine problem amounts to computing $\nu_0(i_I)$.
- Note the optimality condition as follows:

$$\nu_n(i) = \max_{k=1,\ldots,K} \left\{ \sum_{j=1}^{d} M_{ij}^k \nu_{n+1}(j) \right\}.$$

Dynamic Optimization: Backward DP Algorithm

Let $y_{n,i} \in \{1, \ldots, K\}$ be the antibiotic applied in step n and state i.

Algorithm: Backward DP

Set the boundary values as $\nu_N(i) = 1$ for $i = i_F$ and 0 otherwise.
for $n = N-1, \ldots, 0$ **do**
  for $i = 1, \ldots, d$ **do**
    Obtain $\nu_n(i)$ using $\nu_n(i) = \max_{k=1,\ldots,K} \left\{ \sum_{j=1}^{d} M_{ij}^{k} \nu_{n+1}(j) \right\}$ and denote the maximizer as $y_{n,i}$.
Output $y_{n,i}$ and report $\nu_0(i_I)$.

Dynamic Optimization: Backward DP Algorithm

Let $y_{n,i} \in \{1, \ldots, K\}$ be the antibiotic applied in step n and state i.

Algorithm: Backward DP

Set the boundary values as $\nu_N(i) = 1$ for $i = i_F$ and 0 otherwise.
for $n = N-1, \ldots, 0$ **do**
  for $i = 1, \ldots, d$ **do**
    Obtain $\nu_n(i)$ using $\nu_n(i) = \max_{k=1,\ldots,K} \left\{ \sum_{j=1}^{d} M_{ij}^{k} \nu_{n+1}(j) \right\}$ and denote the maximizer as $y_{n,i}$.
Output $y_{n,i}$ and report $\nu_0(i_I)$.

Proposition

The Dynamic Deterministic Version is polynomially solvable in d, K, N.

Proof.

The statement holds true since the running time of the backward DP algorithm is $O(d^2 KN)$. □

Outline

1 Introduction

2 Deterministic Model

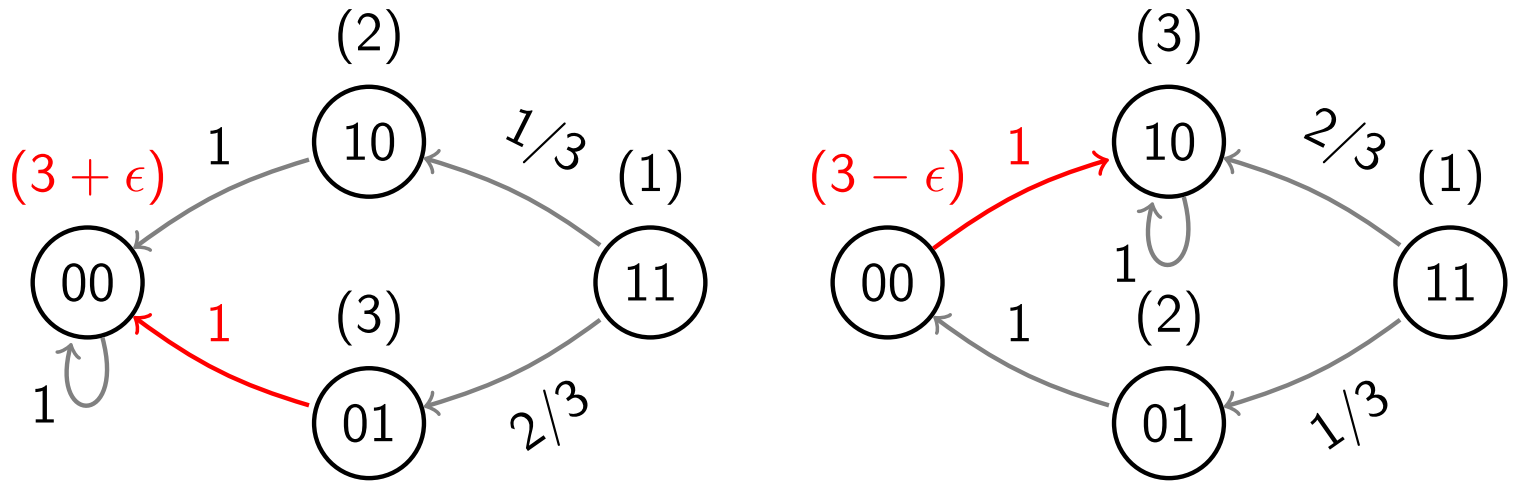
3 Stochastic Model
- An Example
- Uncertainty Modeling and Problem Definition
- Static Optimization
- Dynamic Optimization

4 Computational Results

5 Conclusion

An Example with $a = 2$, $K = 2$ and $N = 2$ (initial $= 11$)

Suppose that growth rate measurements are repeated.
In Replication 2, the growth rate of genotype 00 under antibiotic I (antibiotic II) increases (decreases) to $3 + \epsilon$ ($3 - \epsilon$), for small $\epsilon > 0$.



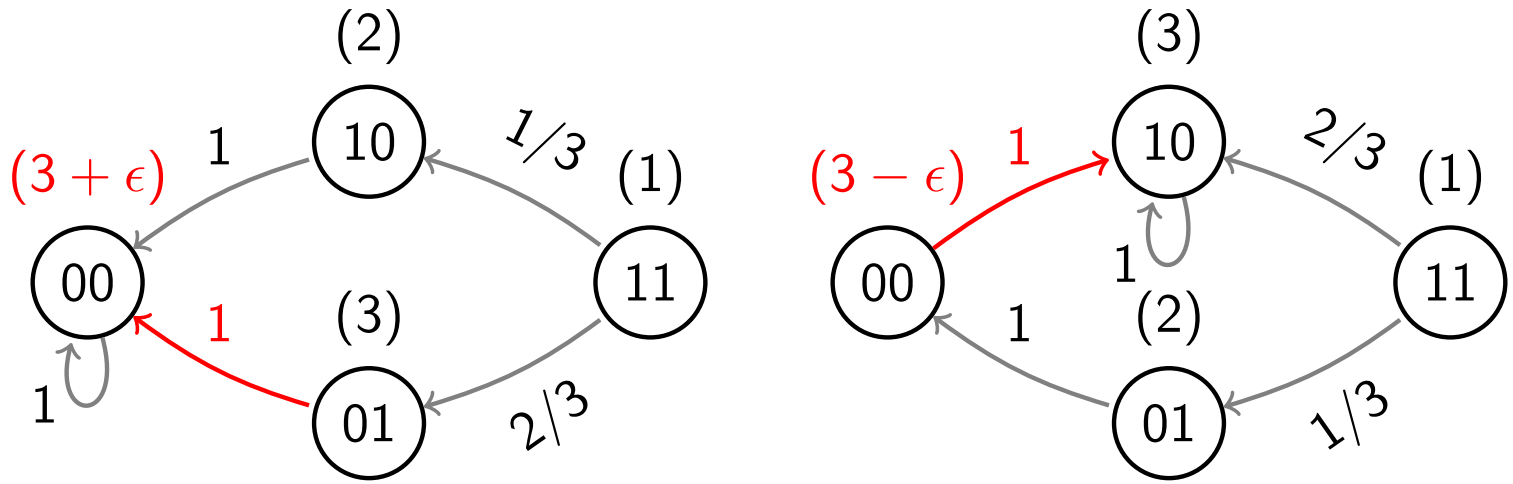
(a) Antibiotic I.

(b) Antibiotic II.

Figure: Markov chains corresponding to two antibiotics in Replication 2.

An Example with $a = 2$, $K = 2$ and $N = 2$ (initial $= 11$)

Suppose that growth rate measurements are repeated.
In Replication 2, the growth rate of genotype 00 under antibiotic I (antibiotic II) increases (decreases) to $3 + \epsilon$ ($3 - \epsilon$), for small $\epsilon > 0$.



(a) Antibiotic I.

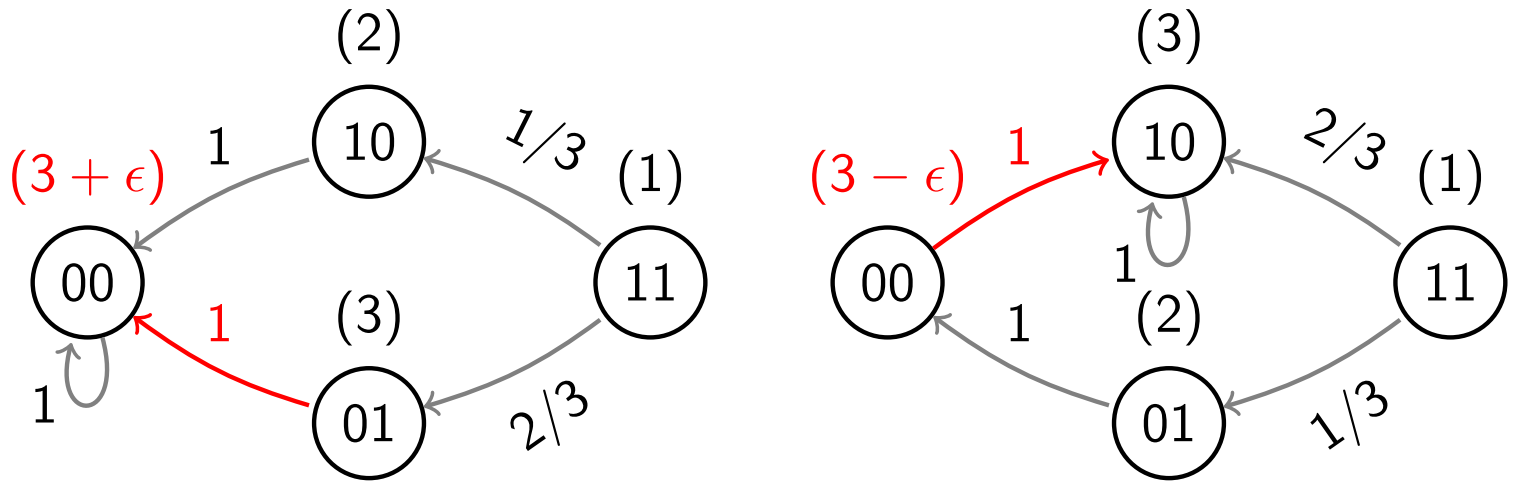
(b) Antibiotic II.

Figure: Markov chains corresponding to two antibiotics in Replication 2.

Static: 4 possible sequences	*Dynamic*: 8 possible policies
Optimal Value: 1	Optimal Value: 1
Optimal Sequence: I-I or II-I	Optimal Policy: I-(I,I), I-(I,II), II-(I,I), II-(I,II)

An Example with $a = 2$, $K = 2$ and $N = 2$ (initial $= 11$)

Suppose that growth rate measurements are repeated.
In Replication 2, the growth rate of genotype 00 under antibiotic I (antibiotic II) increases (decreases) to $3 + \epsilon$ ($3 - \epsilon$), for small $\epsilon > 0$.



(a) Antibiotic I.

(b) Antibiotic II.

Figure: Markov chains corresponding to two antibiotics in Replication 2.

Static: 4 possible sequences	*Dynamic*: 8 possible policies
Optimal Value: 1	Optimal Value: 1
Optimal Sequence: I-I or II-I	Optimal Policy: I-(I,I), I-(I,II), II-(I,I), II-(I,II)

What are the optimal decisions considering BOTH replications?

An Observation

- To solve the Static Stochastic Version, we compute the expected probability of reaching the wild type under each of the four sequences (where the expectation is taken over the replications).
  In this case, the optimal sequence turns out to be II-I with the optimal value of 5/6.

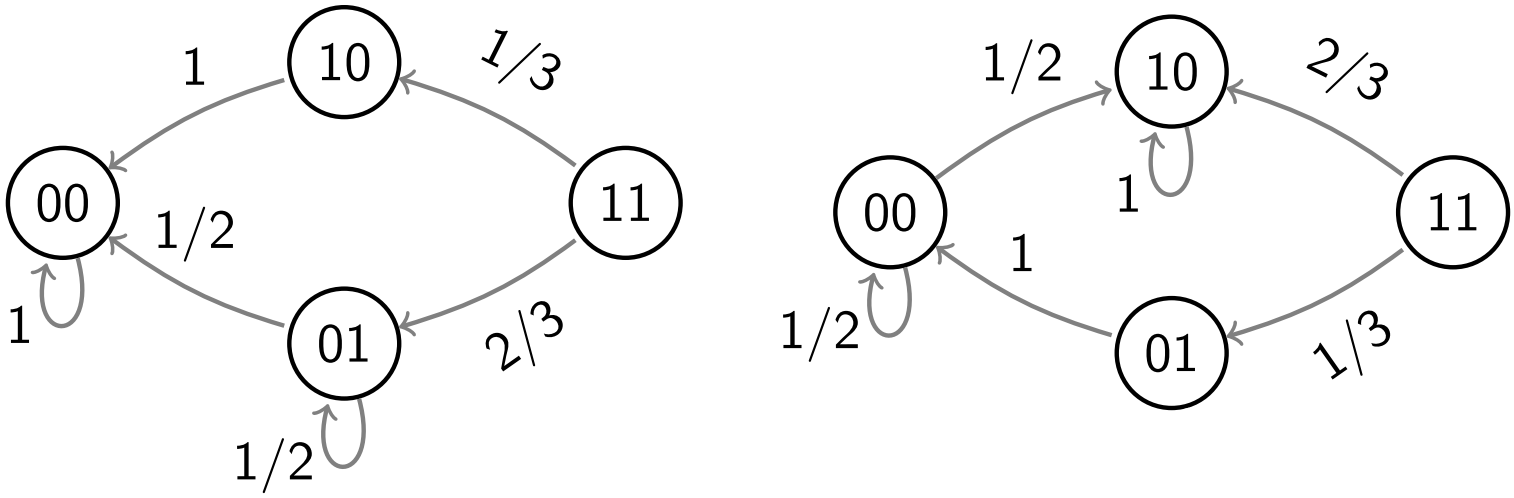
An Observation

- To solve the Static Stochastic Version, we compute the expected probability of reaching the wild type under each of the four sequences (where the expectation is taken over the replications).
  In this case, the optimal sequence turns out to be II-I with the optimal value of 5/6.
- Similarly, to solve the Dynamic Stochastic Version, we can compute the expected probability of reaching the wild type under each of the eight policies.
  In this case, the optimal policies are I-(I,II) and II-(I,II) with the optimal value of 1.

An Observation

- To solve the Static Stochastic Version, we compute the expected probability of reaching the wild type under each of the four sequences (where the expectation is taken over the replications).
  In this case, the optimal sequence turns out to be II-I with the optimal value of 5/6.
- Similarly, to solve the Dynamic Stochastic Version, we can compute the expected probability of reaching the wild type under each of the eight policies.
  In this case, the optimal policies are I-(I,II) and II-(I,II) with the optimal value of 1.
- For this example, the optimal solutions of Static Stochastic and Dynamic Stochastic Versions can be obtained by solving them as particular instances of Static Deterministic and Dynamic Deterministic Versions in which the *average transition probabilities* are used.

An Observation: Average Transition Probabilities



(a) Antibiotic I.

(b) Antibiotic II.

Figure: Average Markov chains corresponding to two antibiotics from two replications.

Static: 4 possible sequences	*Dynamic*: 8 possible policies
Optimal Value: 5/6	Optimal Value: 1
Optimal Sequence: II-I	Optimal Policy: I-(I,II) or II-(I,II)

Uncertainty Modeling

- Suppose that growth rate measurements under each antibiotic are repeated R times.

Uncertainty Modeling

- Suppose that growth rate measurements under each antibiotic are repeated R times.
- To give a perspective, the data acquired from Mira et al. (2017) consists of the growth rates of $d = 16$ genotypes under $K = 23$ different antibiotics (or doses) measured for $R = 12$ times.

Uncertainty Modeling

- Suppose that growth rate measurements under each antibiotic are repeated R times.
- To give a perspective, the data acquired from Mira et al. (2017) consists of the growth rates of $d = 16$ genotypes under $K = 23$ different antibiotics (or doses) measured for $R = 12$ times.
- Let the growth rate of genotype j under the administration of antibiotic k in replication r be given as w_j^{kr}.

Uncertainty Modeling

- Suppose that growth rate measurements under each antibiotic are repeated R times.
- To give a perspective, the data acquired from Mira et al. (2017) consists of the growth rates of $d = 16$ genotypes under $K = 23$ different antibiotics (or doses) measured for $R = 12$ times.
- Let the growth rate of genotype j under the administration of antibiotic k in replication r be given as w_j^{kr}.
- We will assume that the j-th element of the growth rate vector $\boldsymbol{\omega^k} \in \mathbb{R}_+^d$ for antibiotic k is a random variable shown by ω_j^k, and it takes values from the set $\{w_j^{kr} : r = 1, \ldots, R\}$ with equal probability.

Uncertainty Modeling: An Example

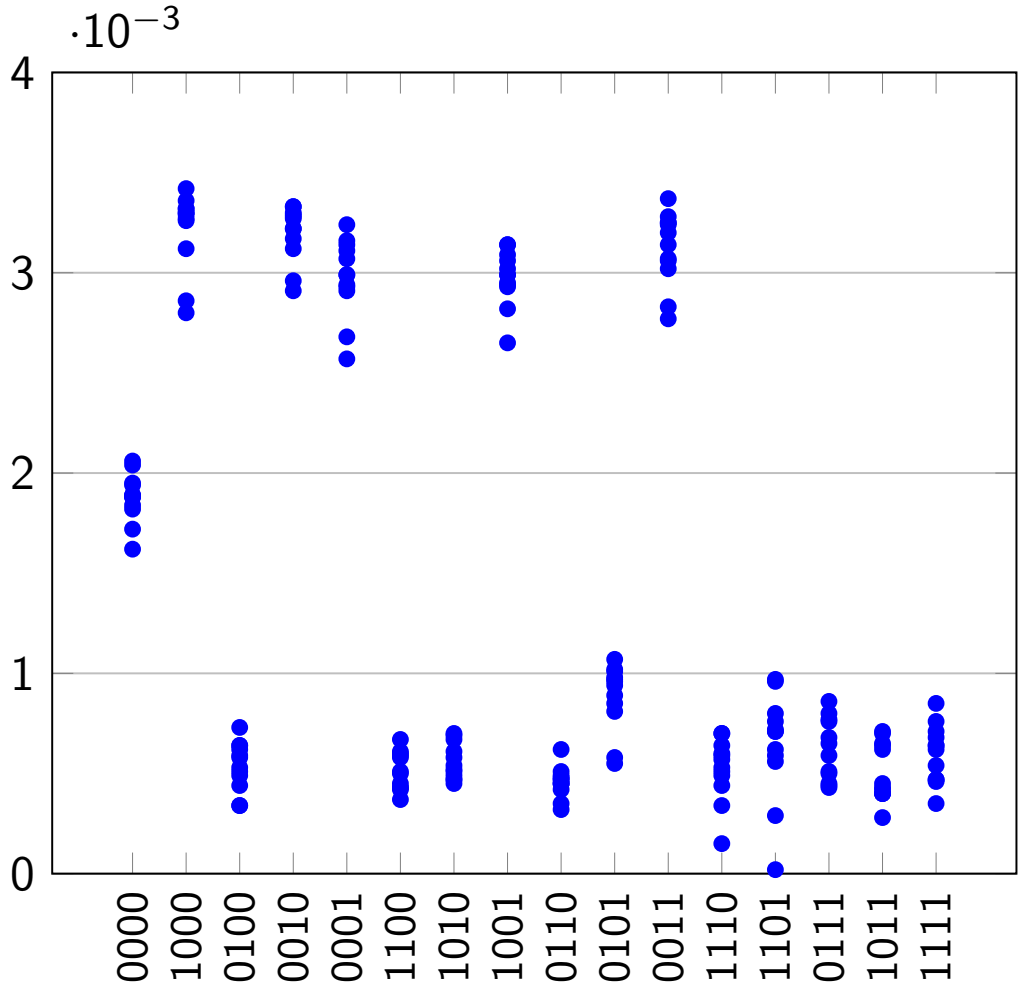


Figure: Growth rate observations of Cefaclor (CEC) with 4 µg/mL concentration (Mira et al. 2017).

Uncertainty Modeling

- To formalize the problem setting, we first define an index set $\mathcal{S} := \{1, \ldots, R\}^d$.

Uncertainty Modeling

- To formalize the problem setting, we first define an index set $\mathcal{S} := \{1, \ldots, R\}^d$.
- For any antibiotic $k = 1, \ldots, K$ and $\boldsymbol{s} \in \mathcal{S}$, we define a vector $\boldsymbol{\omega^{k,s}} \in \mathbb{R}^d$ such that $\omega_j^{k,s} := w_j^{kr}$ if $s_j = r$ for $j = 1, \ldots, d$.

Uncertainty Modeling

- To formalize the problem setting, we first define an index set $\mathcal{S} := \{1, \ldots, R\}^d$.
- For any antibiotic $k = 1, \ldots, K$ and $\boldsymbol{s} \in \mathcal{S}$, we define a vector $\boldsymbol{\omega^{k,s}} \in \mathbb{R}^d$ such that $\omega_j^{k,s} := w_j^{kr}$ if $s_j = r$ for $j = 1, \ldots, d$.
- Notice that if we decide to apply a certain antibiotic, any of R^d many transition probability matrices have an equal probability to be observed.

Uncertainty Modeling

- To formalize the problem setting, we first define an index set $\mathcal{S} := \{1, \ldots, R\}^d$.
- For any antibiotic $k = 1, \ldots, K$ and $\boldsymbol{s} \in \mathcal{S}$, we define a vector $\boldsymbol{\omega^{k,s}} \in \mathbb{R}^d$ such that $\omega_j^{k,s} := w_j^{kr}$ if $s_j = r$ for $j = 1, \ldots, d$.
- Notice that if we decide to apply a certain antibiotic, any of R^d many transition probability matrices have an equal probability to be observed.
- We will denote the matrix for an antibiotic k under the sample $\boldsymbol{s}$ as $\boldsymbol{M^{k,s}}$, which is computed via $C(\boldsymbol{\omega^{k,s}})$ or $E(\boldsymbol{\omega^{k,s}})$ depending on whether CPM or EPM is used.

Problem Definitions

Problem (Static Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Problem Definitions

Problem (Static Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Problem (Dynamic Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *policy* of antibiotics so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Static Optimization

Problem (Static Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Static Optimization

Problem (Static Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *sequence* of antibiotics to be applied so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Nonlinear Formulation:

$$\max \frac{1}{|\mathcal{S}|^N} \sum_{s^1 \in \mathcal{S}} \cdots \sum_{s^N \in \mathcal{S}} \boldsymbol{p}^\top \left(\prod_{n=1}^{N} \boldsymbol{T}_{n,s^n} \right) \boldsymbol{q}$$

$$\text{s.t. } \left(\boldsymbol{T}_{n,s^n}\right)_{s^n \in \mathcal{S}} \in \left\{ \left(\boldsymbol{M}^{k,s}\right)_{s \in \mathcal{S}} : k = 1, \ldots, K \right\} \qquad n = 1, \ldots, N.$$

The Static Stochastic Version *looks* considerably more difficult to solve than the Static Deterministic Version.

Static Optimization: "Equivalence" of Stochastic and Deterministic Versions

- We now prove that the Static Stochastic Version can be formulated as an instance of the Static Deterministic Version.

Static Optimization: "Equivalence" of Stochastic and Deterministic Versions

- We now prove that the Static Stochastic Version can be formulated as an instance of the Static Deterministic Version.
- We will denote the *average transition probability matrix for antibiotic k* by $\bar{\boldsymbol{M}}^{\boldsymbol{k}}$, and compute it as

$$\bar{\boldsymbol{M}}^{\boldsymbol{k}} := \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{s} \in \mathcal{S}} \boldsymbol{M}^{\boldsymbol{k},\boldsymbol{s}}.$$

Static Optimization: "Equivalence" of Stochastic and Deterministic Versions

- We now prove that the Static Stochastic Version can be formulated as an instance of the Static Deterministic Version.
- We will denote the *average transition probability matrix for antibiotic k* by $\bar{\boldsymbol{M}}^{\boldsymbol{k}}$, and compute it as

$$\bar{\boldsymbol{M}}^{\boldsymbol{k}} := \frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{s} \in \mathcal{S}} \boldsymbol{M}^{\boldsymbol{k},\boldsymbol{s}}.$$

Proposition

The Static Stochastic Version is an instance of the Static Deterministic Version.

Static Optimization: Proof of "Equivalence"

Proposition

The Static Stochastic Version is an instance of the Static Deterministic Version.

Proof.

First, we note the following relation:

$$\frac{1}{|\mathcal{S}|^N} \sum_{\boldsymbol{s^1} \in \mathcal{S}} \cdots \sum_{\boldsymbol{s^N} \in \mathcal{S}} \boldsymbol{p}^\top \left(\prod_{n=1}^{N} \boldsymbol{T_{n,s^n}} \right) \boldsymbol{q} = \boldsymbol{p}^\top \prod_{n=1}^{N} \left(\frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{s^n} \in \mathcal{S}} \boldsymbol{T_{n,s^n}} \right) \boldsymbol{q}.$$

In addition, we observe that

$$\frac{1}{|\mathcal{S}|} \sum_{\boldsymbol{s^n} \in \mathcal{S}} \boldsymbol{T_{n,s^n}} \in \{\bar{\boldsymbol{M}}^{\boldsymbol{k}} : k = 1, \ldots, K\},$$

where $\bar{\boldsymbol{M}}^{\boldsymbol{k}}$ are the average transition probability matrices. Therefore, the Static Stochastic Version reduces to an instance of the Static Deterministic Version with $\mathcal{M} = \{\bar{\boldsymbol{M}}^{\boldsymbol{k}} : k = 1, \ldots, K\}$. □

Static Optimization: Solution Approach

- Good News: We can use the MILP of the Static Deterministic Version to solve the Static Stochastic Version.

Static Optimization: Solution Approach

- Good News: We can use the MILP of the Static Deterministic Version to solve the Static Stochastic Version.
- Bad News: The complexity of computing the average transition probability matrices $\bar{\boldsymbol{M}}^{\boldsymbol{k}}$ is $O(R^d)$.

Static Optimization: Solution Approach

- Good News: We can use the MILP of the Static Deterministic Version to solve the Static Stochastic Version.
- Bad News: The complexity of computing the average transition probability matrices $\bar{\boldsymbol{M}}^{\boldsymbol{k}}$ is $O(R^d)$.
  - Resolution: Use sampling to estimate the average transition probability matrices $\bar{\boldsymbol{M}}^{\boldsymbol{k}}$!

Static Optimization: Sample Average Approximation (SAA)

- Let $S \subseteq \mathcal{S}$ be a randomly selected sample index set and compute the following estimator:

$$\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\boldsymbol{S}) := \frac{1}{|S|} \sum_{\boldsymbol{s} \in S} \boldsymbol{M}^{\boldsymbol{k},\boldsymbol{s}}.$$

Static Optimization: Sample Average Approximation (SAA)

- Let $S \subseteq \mathcal{S}$ be a randomly selected sample index set and compute the following estimator:

$$\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\boldsymbol{S}) := \frac{1}{|S|} \sum_{\boldsymbol{s} \in S} \boldsymbol{M}^{\boldsymbol{k},\boldsymbol{s}}.$$

- We use SAA and solve the MILP model with estimated matrices.

Static Optimization: Sample Average Approximation (SAA)

- Let $S \subseteq \mathcal{S}$ be a randomly selected sample index set and compute the following estimator:

$$\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\boldsymbol{S}) := \frac{1}{|S|} \sum_{\boldsymbol{s} \in S} \boldsymbol{M}^{\boldsymbol{k},\boldsymbol{s}}.$$

- We use SAA and solve the MILP model with estimated matrices.
- Given samples $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$, we solve the MILP with $\mathcal{M} = \{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$ and compute

$$z_{\text{opt}}(\dot{S}^1, \ldots, \dot{S}^K) := \boldsymbol{p}^\top \left(\prod_{n=1}^{N} \dot{\boldsymbol{T}}_{\boldsymbol{n}} \right) \boldsymbol{q}, \quad \text{where } \dot{\boldsymbol{T}}_{\boldsymbol{n}} = \hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) \text{ if } x_{n,k} = 1.$$

Static Optimization: Sample Average Approximation (SAA)

- Let $S \subseteq \mathcal{S}$ be a randomly selected sample index set and compute the following estimator:

$$\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\boldsymbol{S}) := \frac{1}{|S|} \sum_{\boldsymbol{s} \in S} \boldsymbol{M}^{\boldsymbol{k},\boldsymbol{s}}.$$

- We use SAA and solve the MILP model with estimated matrices.
- Given samples $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$, we solve the MILP with $\mathcal{M} = \{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$ and compute

$$z_{\text{opt}}(\dot{S}^1, \ldots, \dot{S}^K) := \boldsymbol{p}^\top \left(\prod_{n=1}^{N} \dot{\boldsymbol{T}}_{\boldsymbol{n}} \right) \boldsymbol{q}, \quad \text{where } \dot{\boldsymbol{T}}_{\boldsymbol{n}} = \hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) \text{ if } x_{n,k} = 1.$$

- We adopt an out-of-sample evaluation scheme to test the success of our solution: Generate a new sample $\ddot{S}^1, \ldots, \ddot{S}^K \subseteq \mathcal{S}$ and evaluate

$$z_{\text{ev}}(\ddot{S}^1, \ldots, \ddot{S}^K) := \boldsymbol{p}^\top \left(\prod_{n=1}^{N} \ddot{\boldsymbol{T}}_{\boldsymbol{n}} \right) \boldsymbol{q}, \quad \text{where } \ddot{\boldsymbol{T}}_{\boldsymbol{n}} = \hat{\boldsymbol{M}}^{\boldsymbol{k}}(\ddot{\boldsymbol{S}}^{\boldsymbol{k}}).$$

Dynamic Optimization

Problem (Dynamic Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *policy* of antibiotics so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Dynamic Optimization

Problem (Dynamic Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *policy* of antibiotics so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Proposition

The Dynamic Stochastic Version is an instance of the Dynamic Deterministic Version.

Dynamic Optimization

Problem (Dynamic Stochastic Version)

Given K antibiotics with their uncertain but equally likely transition probability matrices $\{\boldsymbol{M}^{k,s} : s \in \mathcal{S}\}$, $k = 1, \ldots, K$, the length of the treatment plan N and the initial distribution $\boldsymbol{p}$, determine a *policy* of antibiotics so that the *expected* probability of reaching the desired final distribution $\boldsymbol{q}$ is maximized.

Proposition

The Dynamic Stochastic Version is an instance of the Dynamic Deterministic Version.

Good News: We can use the DP Algorithm of the Dynamic Deterministic Version to solve the Dynamic Stochastic Version.

- We still need to use "SAA"!

Dynamic Optimization: Solution Approach

- Given samples $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$, we first compute the set of estimator matrices $\mathcal{M} = \{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$.

Dynamic Optimization: Solution Approach

- Given samples $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$, we first compute the set of estimator matrices $\mathcal{M} = \{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$.
- We then use the DP approach to obtain a policy, which we will denote by $y_{n,i}$.

Dynamic Optimization: Solution Approach

- Given samples $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$, we first compute the set of estimator matrices $\mathcal{M} = \{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$.
- We then use the DP approach to obtain a policy, which we will denote by $y_{n,i}$.
- Let us denote the optimal value with respect to the sample $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$ as z_{opt} again.

Dynamic Optimization: Solution Approach

- Given samples $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$, we first compute the set of estimator matrices $\mathcal{M} = \{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$.
- We then use the DP approach to obtain a policy, which we will denote by $y_{n,i}$.
- Let us denote the optimal value with respect to the sample $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$ as z_{opt} again.
- We adopt an out-of-sample evaluation scheme to test the success of our solution: Generate a new sample $\ddot{S}^1, \ldots, \ddot{S}^K \subseteq \mathcal{S}$ and compute the estimator matrices as $\{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\ddot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$.

Dynamic Optimization: Solution Approach

- Given samples $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$, we first compute the set of estimator matrices $\mathcal{M} = \{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\dot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$.
- We then use the DP approach to obtain a policy, which we will denote by $y_{n,i}$.
- Let us denote the optimal value with respect to the sample $\dot{S}^1, \ldots, \dot{S}^K \subseteq \mathcal{S}$ as z_{opt} again.
- We adopt an out-of-sample evaluation scheme to test the success of our solution: Generate a new sample $\ddot{S}^1, \ldots, \ddot{S}^K \subseteq \mathcal{S}$ and compute the estimator matrices as $\{\hat{\boldsymbol{M}}^{\boldsymbol{k}}(\ddot{\boldsymbol{S}}^{\boldsymbol{k}}) : k = 1, \ldots, K\}$.
- We use a forward DP algorithm to evaluate the objective function value of the policy with respect to the sample $\ddot{S}^1, \ldots, \ddot{S}^K \subseteq \mathcal{S}$, which we will denote as z_{ev} again.

Dynamic Optimization: Forward DP Algorithm

Algorithm: Forward DP

Set $\nu_0(i_I) = 1$ and $\nu_0(i) = 0$ for $i \neq i_I$.
for $n = 1, \ldots, N$ **do**
  for $i = 1, \ldots, d$ **do**
    Let $k = y_{n,i}$ and $\hat{\boldsymbol{M}} = \hat{\boldsymbol{M}}^{\boldsymbol{k}}(\ddot{\boldsymbol{S}}^{\boldsymbol{k}})$.
    Compute

$$\nu_n(j) = \sum_{i=1}^{d} \hat{M}_{ij} \nu_{n-1}(i).$$

Output $\nu_N(i_F)$.

The running time of the Forward DP Algorithm is $O(d^2 N)$.

Outline

1 Introduction

2 Deterministic Model

3 Stochastic Model

4 Computational Results
- Computational Setting
- Results of the Static Stochastic Version
- Results of the Dynamic Stochastic Version

5 Conclusion

Computational Setting

- We use the experimental growth rate data from Mira et al. (2017) to obtain the transition probability matrices ($d = 16$ genotypes, $K = 23$ different antibiotics/doses, $R = 12$ replications).

Computational Setting

- We use the experimental growth rate data from Mira et al. (2017) to obtain the transition probability matrices ($d = 16$ genotypes, $K = 23$ different antibiotics/doses, $R = 12$ replications).
- We set $|\dot{S}^k| = |\ddot{S}^k| = 10^4$ to estimate the average transition probability matrices (out of R^d).

Computational Setting

- We use the experimental growth rate data from Mira et al. (2017) to obtain the transition probability matrices ($d = 16$ genotypes, $K = 23$ different antibiotics/doses, $R = 12$ replications).
- We set $|\dot{S}^k| = |\ddot{S}^k| = 10^4$ to estimate the average transition probability matrices (out of R^d).
- We use a 64-bit workstation with two Intel(R) Xeon(R) Gold 6248R CPU (3.00GHz) processors (256 GB RAM) using the Python programming language.

Computational Setting

- We use the experimental growth rate data from Mira et al. (2017) to obtain the transition probability matrices ($d = 16$ genotypes, $K = 23$ different antibiotics/doses, $R = 12$ replications).
- We set $|\dot{S}^k| = |\ddot{S}^k| = 10^4$ to estimate the average transition probability matrices (out of R^d).
- We use a 64-bit workstation with two Intel(R) Xeon(R) Gold 6248R CPU (3.00GHz) processors (256 GB RAM) using the Python programming language.
- We utilize Gurobi 9.5 to solve the MILP of the Static Stochastic Version with the default settings (except the absolute optimality gap parameter is set to 0.001, and the time limit is set to 1 hr).

Computational Setting

- We use the experimental growth rate data from Mira et al. (2017) to obtain the transition probability matrices ($d = 16$ genotypes, $K = 23$ different antibiotics/doses, $R = 12$ replications).
- We set $|\dot{S}^k| = |\ddot{S}^k| = 10^4$ to estimate the average transition probability matrices (out of R^d).
- We use a 64-bit workstation with two Intel(R) Xeon(R) Gold 6248R CPU (3.00GHz) processors (256 GB RAM) using the Python programming language.
- We utilize Gurobi 9.5 to solve the MILP of the Static Stochastic Version with the default settings (except the absolute optimality gap parameter is set to 0.001, and the time limit is set to 1 hr).
- We utilize the DP approach to solve the Dynamic Stochastic Version.

Computational Setting

- We use the experimental growth rate data from Mira et al. (2017) to obtain the transition probability matrices ($d = 16$ genotypes, $K = 23$ different antibiotics/doses, $R = 12$ replications).
- We set $|\dot{S}^k| = |\ddot{S}^k| = 10^4$ to estimate the average transition probability matrices (out of R^d).
- We use a 64-bit workstation with two Intel(R) Xeon(R) Gold 6248R CPU (3.00GHz) processors (256 GB RAM) using the Python programming language.
- We utilize Gurobi 9.5 to solve the MILP of the Static Stochastic Version with the default settings (except the absolute optimality gap parameter is set to 0.001, and the time limit is set to 1 hr).
- We utilize the DP approach to solve the Dynamic Stochastic Version.
- We add the $d \times d$ identity matrix as the $(K+1)$-st antibiotic matrix to model the "no intake" action so that the optimal values are made monotonically nondecreasing in the number of steps N.

Static Stochastic Version–CPM

i_I: initial genotype "AT(s)": average MILP CPU time in seconds

	$N=4$		$N=8$		$N=12$		$N=16$	
i_I	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.49	0.49	0.66	0.66	0.71	0.71	0.73	0.73
0010	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77
0011	0.72	0.72	0.74	0.74	0.75	0.75	0.75	0.75
0100	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94
0101	0.54	0.54	0.69	0.69	0.72	0.72	0.73	0.73
0110	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77
0111	0.61	0.61	0.73	0.73	0.74	0.74	0.74	0.75
1000	0.63	0.63	0.63	0.63	0.67	0.67	0.70	0.71
1001	0.57	0.57	0.57	0.57	0.65	0.66	0.70	0.70
1010	0.70	0.70	0.71	0.71	0.72	0.72	0.73	0.73
1011	0.48	0.48	0.66	0.66	0.70	0.70	0.72	0.73
1100	0.60	0.60	0.60	0.60	0.66	0.66	0.70	0.71
1101	0.57	0.57	0.57	0.57	0.66	0.66	0.70	0.70
1110	0.64	0.64	0.68	0.68	0.71	0.71	0.73	0.73
1111	0.52	0.52	0.58	0.58	0.66	0.67	0.70	0.70
AT(s)	0.296		8.164		168.888		2655.636	

Static Stochastic Version–CPM

i_I: initial genotype "AT(s)": average MILP CPU time in seconds

	$N=4$		$N=8$		$N=12$		$N=16$	
i_I	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.49	0.49	0.66	0.66	0.71	0.71	0.73	0.73
0010	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77
0011	0.72	0.72	0.74	0.74	0.75	0.75	0.75	0.75
0100	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94
0101	0.54	0.54	0.69	0.69	0.72	0.72	0.73	0.73
0110	0.77	0.77	0.77	0.77	0.77	0.77	0.77	0.77
0111	0.61	0.61	0.73	0.73	0.74	0.74	0.74	0.75
1000	0.63	0.63	0.63	0.63	0.67	0.67	0.70	0.71
1001	0.57	0.57	0.57	0.57	0.65	0.66	0.70	0.70
1010	0.70	0.70	0.71	0.71	0.72	0.72	0.73	0.73
1011	0.48	0.48	0.66	0.66	0.70	0.70	0.72	0.73
1100	0.60	0.60	0.60	0.60	0.66	0.66	0.70	0.71
1101	0.57	0.57	0.57	0.57	0.66	0.66	0.70	0.70
1110	0.64	0.64	0.68	0.68	0.71	0.71	0.73	0.73
1111	0.52	0.52	0.58	0.58	0.66	0.67	0.70	0.70
AT(s)	0.296		8.164		168.888		2655.636	

- SAA-based approach is **very accurate**: $|z_{opt} - z_{ev}| \leq 0.01$
- The probability of reaching the wild type exceeds 0.695 for all i_I when $N = 16$.
- CPU time increases rapidly (some time outs when $N \geq 14$).

Static Stochastic Version–EPM

	$N = 4$		$N = 8$		$N = 12$		$N = 16$	
i_l	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.46	0.46	0.49	0.49	0.49	0.49	0.50	0.50
0010	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57
0011	0.47	0.47	0.48	0.48	0.49	0.49	0.50	0.50
0100	0.51	0.50	0.51	0.50	0.51	0.50	0.51	0.50
0101	0.42	0.42	0.49	0.49	0.50	0.50	0.51	0.50
0110	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53
0111	0.41	0.40	0.48	0.48	0.50	0.49	0.50	0.50
1000	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62
1001	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54
1010	0.47	0.48	0.49	0.49	0.49	0.49	0.50	0.50
1011	0.39	0.39	0.49	0.49	0.50	0.49	0.50	0.50
1100	0.49	0.49	0.53	0.53	0.53	0.53	0.53	0.53
1101	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54
1110	0.31	0.31	0.50	0.50	0.50	0.50	0.51	0.50
1111	0.44	0.44	0.51	0.51	0.52	0.51	0.52	0.51
AT(s)	0.320		5.915		90.005		1803.956	

Static Stochastic Version–EPM

	$N = 4$		$N = 8$		$N = 12$		$N = 16$	
i_l	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.46	0.46	0.49	0.49	0.49	0.49	0.50	0.50
0010	0.57	0.57	0.57	0.57	0.57	0.57	0.57	0.57
0011	0.47	0.47	0.48	0.48	0.49	0.49	0.50	0.50
0100	0.51	0.50	0.51	0.50	0.51	0.50	0.51	0.50
0101	0.42	0.42	0.49	0.49	0.50	0.50	0.51	0.50
0110	0.53	0.53	0.53	0.53	0.53	0.53	0.53	0.53
0111	0.41	0.40	0.48	0.48	0.50	0.49	0.50	0.50
1000	0.62	0.62	0.62	0.62	0.62	0.62	0.62	0.62
1001	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54
1010	0.47	0.48	0.49	0.49	0.49	0.49	0.50	0.50
1011	0.39	0.39	0.49	0.49	0.50	0.49	0.50	0.50
1100	0.49	0.49	0.53	0.53	0.53	0.53	0.53	0.53
1101	0.54	0.54	0.54	0.54	0.54	0.54	0.54	0.54
1110	0.31	0.31	0.50	0.50	0.50	0.50	0.51	0.50
1111	0.44	0.44	0.51	0.51	0.52	0.51	0.52	0.51
AT(s)	0.320		5.915		90.005		1803.956	

- SAA-based approach is **very accurate**: $|z_{\text{opt}} - z_{\text{ev}}| \leq 0.01$
- The probability of reaching the wild type exceeds 0.495 for all i_l when $N = 16$.
- CPU time also increases (some time outs when $N = 16$).

Dynamic Stochastic Version–CPM

	$N = 4$		$N = 8$		$N = 11$	
i_I	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.86	0.86	0.99	0.99	1.00	1.00
0010	0.95	0.95	1.00	1.00	1.00	1.00
0011	0.90	0.90	0.99	0.99	1.00	1.00
0100	0.98	0.98	1.00	1.00	1.00	1.00
0101	0.77	0.77	0.98	0.98	1.00	1.00
0110	0.94	0.94	1.00	1.00	1.00	1.00
0111	0.77	0.78	0.98	0.98	1.00	1.00
1000	0.95	0.95	1.00	1.00	1.00	1.00
1001	0.86	0.86	0.99	0.99	1.00	1.00
1010	0.91	0.91	0.99	0.99	1.00	1.00
1011	0.72	0.72	0.98	0.98	1.00	1.00
1100	0.89	0.89	0.99	0.99	1.00	1.00
1101	0.73	0.73	0.98	0.98	1.00	1.00
1110	0.78	0.78	0.98	0.99	1.00	1.00
1111	0.62	0.62	0.97	0.97	1.00	1.00

Dynamic Stochastic Version–CPM

	$N = 4$		$N = 8$		$N = 11$	
i_l	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.86	0.86	0.99	0.99	1.00	1.00
0010	0.95	0.95	1.00	1.00	1.00	1.00
0011	0.90	0.90	0.99	0.99	1.00	1.00
0100	0.98	0.98	1.00	1.00	1.00	1.00
0101	0.77	0.77	0.98	0.98	1.00	1.00
0110	0.94	0.94	1.00	1.00	1.00	1.00
0111	0.77	0.78	0.98	0.98	1.00	1.00
1000	0.95	0.95	1.00	1.00	1.00	1.00
1001	0.86	0.86	0.99	0.99	1.00	1.00
1010	0.91	0.91	0.99	0.99	1.00	1.00
1011	0.72	0.72	0.98	0.98	1.00	1.00
1100	0.89	0.89	0.99	0.99	1.00	1.00
1101	0.73	0.73	0.98	0.98	1.00	1.00
1110	0.78	0.78	0.98	0.99	1.00	1.00
1111	0.62	0.62	0.97	0.97	1.00	1.00

- SAA-based approach is **very accurate**: $|z_{\text{opt}} - z_{\text{ev}}| \leq 0.01$
- The probability of reaching the wild type exceeds 0.995 for all i_l when $N = 11$.
- CPU time of DP is negligible.

Dynamic Stochastic Version–EPM

	$N = 4$		$N = 8$		$N = 12$		$N = 14$	
i_I	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.82	0.82	0.97	0.97	1.00	1.00	1.00	1.00
0010	0.91	0.91	0.99	0.99	1.00	1.00	1.00	1.00
0011	0.77	0.77	0.96	0.97	0.99	1.00	1.00	1.00
0100	0.78	0.78	0.97	0.97	1.00	1.00	1.00	1.00
0101	0.69	0.69	0.95	0.95	0.99	0.99	1.00	1.00
0110	0.83	0.84	0.97	0.98	1.00	1.00	1.00	1.00
0111	0.61	0.61	0.94	0.95	0.99	0.99	1.00	1.00
1000	0.94	0.94	0.99	0.99	1.00	1.00	1.00	1.00
1001	0.83	0.83	0.98	0.98	1.00	1.00	1.00	1.00
1010	0.80	0.81	0.97	0.97	1.00	1.00	1.00	1.00
1011	0.59	0.59	0.95	0.95	0.99	0.99	1.00	1.00
1100	0.71	0.70	0.96	0.96	0.99	0.99	1.00	1.00
1101	0.70	0.70	0.96	0.96	1.00	1.00	1.00	1.00
1110	0.52	0.52	0.94	0.94	0.99	0.99	1.00	1.00
1111	0.48	0.48	0.93	0.93	0.99	0.99	1.00	1.00

Dynamic Stochastic Version–EPM

	$N = 4$		$N = 8$		$N = 12$		$N = 14$	
i_l	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}	z_{opt}	z_{ev}
0001	0.82	0.82	0.97	0.97	1.00	1.00	1.00	1.00
0010	0.91	0.91	0.99	0.99	1.00	1.00	1.00	1.00
0011	0.77	0.77	0.96	0.97	0.99	1.00	1.00	1.00
0100	0.78	0.78	0.97	0.97	1.00	1.00	1.00	1.00
0101	0.69	0.69	0.95	0.95	0.99	0.99	1.00	1.00
0110	0.83	0.84	0.97	0.98	1.00	1.00	1.00	1.00
0111	0.61	0.61	0.94	0.95	0.99	0.99	1.00	1.00
1000	0.94	0.94	0.99	0.99	1.00	1.00	1.00	1.00
1001	0.83	0.83	0.98	0.98	1.00	1.00	1.00	1.00
1010	0.80	0.81	0.97	0.97	1.00	1.00	1.00	1.00
1011	0.59	0.59	0.95	0.95	0.99	0.99	1.00	1.00
1100	0.71	0.70	0.96	0.96	0.99	0.99	1.00	1.00
1101	0.70	0.70	0.96	0.96	1.00	1.00	1.00	1.00
1110	0.52	0.52	0.94	0.94	0.99	0.99	1.00	1.00
1111	0.48	0.48	0.93	0.93	0.99	0.99	1.00	1.00

- SAA-based approach is **very accurate**: $|z_{\text{opt}} - z_{\text{ev}}| \leq 0.01$
- The probability of reaching the wild type exceeds 0.995 for all i_l when $N = 14$.
- CPU time of DP is negligible.

Outline

1 Introduction

2 Deterministic Model

3 Stochastic Model

4 Computational Results

5 Conclusion
- Ongoing Work

Ongoing Work

We are currently working on the following extensions/versions:

Ongoing Work

We are currently working on the following extensions/versions:

- Intermediate solutions between static and dynamic versions
  - What if we observe state transitions in every m steps?
  - What would be the optimal observation points?

Ongoing Work

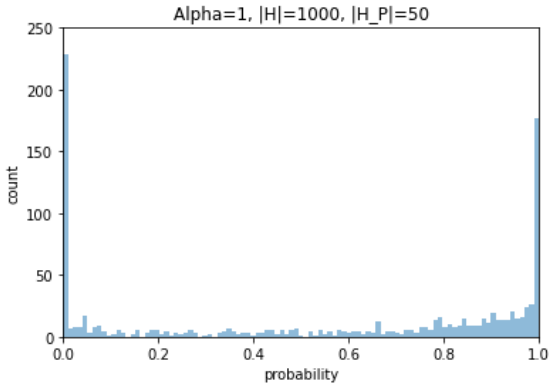
We are currently working on the following extensions/versions:

- Intermediate solutions between static and dynamic versions
  - What if we observe state transitions in every m steps?
  - What would be the optimal observation points?
- Risk-averse solutions
  - What if we change the objective function to maximizing the average of worst $p\%$ of the realizations?
  - What would be the trade-off between expected performance and risk?

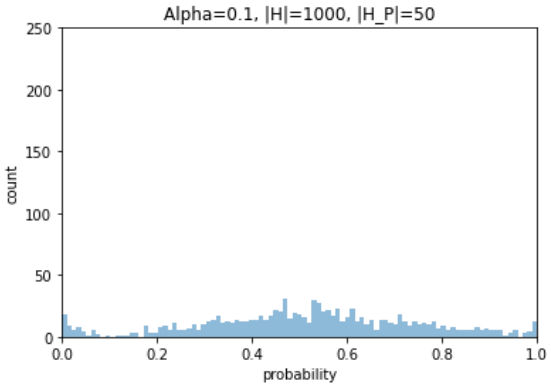
Back-up Slides

Results from an Out-of-Sample Analysis - CPM

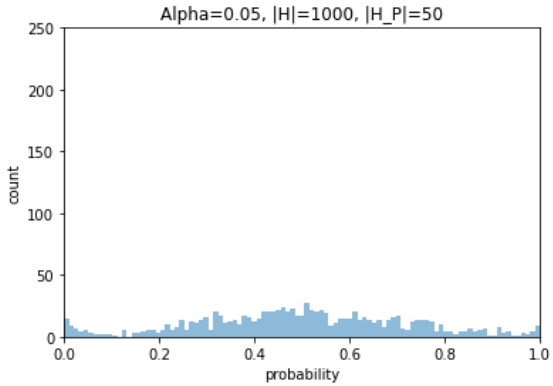
The initial genotype is 1111, $N = 4$, the probability model is CPM, $\alpha = 0.05, 0.10$. We set $|H| = |H'| = 1000$, $|B| = 20$ and $|H_b| = 50$.



Risk Neutral



Risk Averse $\alpha = 0.10$



Risk Averse $\alpha = 0.05$

	Risk Neut.	Risk Av. $\alpha = 0.10$	Risk Av. $\alpha = 0.05$
Average	0.56	0.52	0.50
Av.Worst 10%	0.00	0.10	0.11
Av.Worst 5%	0.00	0.02	0.03

Outline

- Step 1: Bilinearization
- Step 2: Linearization
- Optimization Model
- Computational Results
- Results–Randomly Generated Instances

Bilinearization Step

$$\max_{T,w} f(w)$$
$$\text{s.t. } pT_1 \cdots T_N = w$$
$$T_1, \ldots, T_N \in \mathcal{T}.$$

- Let us define the recursion $u_n = u_{n-1} T_n$ with $u_0 := p \quad (u_n \in \mathbb{C}^{r \times d})$.

Bilinearization Step

$$\begin{aligned} \max_{T,w} \; & f(w) \\ \text{s.t. } & pT_1 \cdots T_N = w \\ & T_1, \ldots, T_N \in \mathcal{T}. \end{aligned}$$

- Let us define the recursion $u_n = u_{n-1} T_n$ with $u_0 := p \quad (u_n \in \mathbb{C}^{r \times d})$.

$$\begin{aligned} u_1 &= u_0 T_1 = pT_1 \\ u_2 &= u_1 T_2 = pT_1 T_2 \\ &\vdots \\ u_N &= u_{N-1} T_N = pT_1 \cdots T_N = w. \end{aligned}$$

Bilinearization Step

$$\begin{aligned} \max_{T,w} \; & f(w) \\ \text{s.t. } & pT_1 \cdots T_N = w \\ & T_1, \ldots, T_N \in \mathcal{T}. \end{aligned}$$

- Let us define the recursion $u_n = u_{n-1} T_n$ with $u_0 := p \quad (u_n \in \mathbb{C}^{r \times d})$.

$$\begin{aligned} u_1 &= u_0 T_1 = pT_1 \\ u_2 &= u_1 T_2 = pT_1 T_2 \\ &\vdots \\ u_N &= u_{N-1} T_N = pT_1 \cdots T_N = w. \end{aligned}$$

- We then reformulate the problem as follows:

$$\begin{aligned} \max_{T,u} \; & f(u_N) \\ \text{s.t. } & u_0 = p \\ & u_{n-1} T_n = u_n \qquad n = 1, \ldots, N \\ & T_1, \ldots, T_N \in \mathcal{T}. \end{aligned}$$

Bilinearization Step: An Observation

$$\begin{aligned} \max_{T,u} \; & f(u_N) \\ \text{s.t. } & u_0 = p \\ & u_{n-1} T_n = u_n \qquad n = 1, \ldots, N \\ & T_1, \ldots, T_N \in \mathcal{T}. \end{aligned}$$

- We note that, by construction, each u_n matrix is guaranteed to come from a "polytope" $\mathcal{U}_n \subseteq \mathbb{C}^{r \times d}$ defined as

$$\mathcal{U}_n := \text{conv}\left(\bigcup_{k=1}^{K} \{ u_{n-1} M^k : u_{n-1} \in \mathcal{U}_{n-1} \} \right), \; n = 1, \ldots, n$$

with $\mathcal{U}_0 := \{p\}$.

Bilinearization Step: An Observation

$$\begin{aligned} \max_{T,u} \; & f(u_N) \\ \text{s.t. } & u_0 = p \\ & u_{n-1} T_n = u_n \qquad n = 1, \ldots, N \\ & T_1, \ldots, T_N \in \mathcal{T}. \end{aligned}$$

- We note that, by construction, each u_n matrix is guaranteed to come from a "polytope" $\mathcal{U}_n \subseteq \mathbb{C}^{r \times d}$ defined as

$$\mathcal{U}_n := \text{conv}\left(\bigcup_{k=1}^{K} \{ u_{n-1} M^k : u_{n-1} \in \mathcal{U}_{n-1} \} \right), \; n = 1, \ldots, n$$

  with $\mathcal{U}_0 := \{p\}$.

- This observation will be crucial in the linearization step.

Linearization Step

$$\begin{aligned} \max_{T,u} \; & f(u_N) \\ \text{s.t.} \; & u_0 = p \\ & u_{n-1} T_n = u_n \qquad n = 1, \ldots, N \\ & T_1, \ldots, T_N \in \mathcal{T}. \end{aligned}$$

- After the Bilinearization Step, we still have bilinear and combinatorial constraints (not a tractable form).

Linearization Step

$$
\begin{aligned}
\max_{T,u} \; & f(u_N) \\
\text{s.t.} \; & u_0 = p \\
& u_{n-1} T_n = u_n \qquad n = 1, \ldots, N \\
& T_1, \ldots, T_N \in \mathcal{T}.
\end{aligned}
$$

- After the Bilinearization Step, we still have bilinear and combinatorial constraints (not a tractable form).
- We now use disjunctive arguments (Balas, 1979) to obtain a mixed-integer linear representation of the feasible region.

Linearization Step: A Proposition

Proposition

Given $\mathcal{T} = \{M^k\}_{k=1}^K \subseteq \mathbb{C}^{d\times d}$ *and a polytope* $\mathcal{Y} \subseteq \mathbb{C}^{r\times d}$, *consider*

$$\mathcal{S} := \left\{(y, T, z) \in \mathcal{Y} \times \mathcal{T} \times \mathbb{C}^{r\times d} : yT = z\right\}.$$

Linearization Step: A Proposition

Proposition

Given $\mathcal{T} = \{M^k\}_{k=1}^K \subseteq \mathbb{C}^{d\times d}$ *and a polytope* $\mathcal{Y} \subseteq \mathbb{C}^{r\times d}$, *consider*

$$\mathcal{S} := \left\{(y, T, z) \in \mathcal{Y} \times \mathcal{T} \times \mathbb{C}^{r\times d} : yT = z\right\}.$$

1. *The following in* (y, T, z, v_k, μ_k) *is an extended formulation for* $\mathcal{S}$:

$$\sum_{k=1}^K \mu_k M^k = T, \quad \sum_{k=1}^K \mu_k = 1 \tag{I}$$

$$\sum_{k=1}^K v_k M^k = z, \quad \sum_{k=1}^K v_k = y \tag{II}$$

$$v_k \in \mathcal{Y}\mu_k, \quad \mu_k \in \{0,1\}, \ k = 1, \ldots, K. \tag{III}$$

Linearization Step: A Proposition

Proposition

Given $\mathcal{T} = \{M^k\}_{k=1}^K \subseteq \mathbb{C}^{d\times d}$ *and a polytope* $\mathcal{Y} \subseteq \mathbb{C}^{r\times d}$, *consider*

$$\mathcal{S} := \left\{(y, T, z) \in \mathcal{Y} \times \mathcal{T} \times \mathbb{C}^{r\times d} : yT = z\right\}.$$

1. *The following in* (y, T, z, v_k, μ_k) *is an extended formulation for* $\mathcal{S}$*:*

$$\sum_{k=1}^K \mu_k M^k = T, \quad \sum_{k=1}^K \mu_k = 1 \tag{I}$$

$$\sum_{k=1}^K v_k M^k = z, \quad \sum_{k=1}^K v_k = y \tag{II}$$

$$v_k \in \mathcal{Y}\mu_k, \quad \mu_k \in \{0,1\}, \ k = 1, \ldots, K. \tag{III}$$

2. $\text{conv}(\mathcal{S}) = \left\{(y, T, z) \in \mathcal{Y} \times \mathcal{T} \times \mathbb{C}^{r\times d} : \ \exists (v_k, \mu_k) \in \mathcal{Y}\mu_k \times \mathbb{R}_+ : (\text{I}) - (\text{II})\right\}.$

Linearization Step: A Proposition

Proposition

Given $\mathcal{T} = \{M^k\}_{k=1}^K \subseteq \mathbb{C}^{d\times d}$ *and a polytope* $\mathcal{Y} \subseteq \mathbb{C}^{r\times d}$, *consider*

$$\mathcal{S} := \left\{(y, T, z) \in \mathcal{Y} \times \mathcal{T} \times \mathbb{C}^{r\times d} : yT = z\right\}.$$

1. *The following in* (y, T, z, v_k, μ_k) *is an extended formulation for* $\mathcal{S}$:

$$\sum_{k=1}^K \mu_k M^k = T, \quad \sum_{k=1}^K \mu_k = 1 \tag{I}$$

$$\sum_{k=1}^K v_k M^k = z, \quad \sum_{k=1}^K v_k = y \tag{II}$$

$$v_k \in \mathcal{Y}\mu_k, \quad \mu_k \in \{0,1\}, \ k = 1, \ldots, K. \tag{III}$$

2. $\text{conv}(\mathcal{S}) = \left\{(y, T, z) \in \mathcal{Y} \times \mathcal{T} \times \mathbb{C}^{r\times d} : \ \exists (v_k, \mu_k) \in \mathcal{Y}\mu_k \times \mathbb{R}_+ : (\text{I}) - (\text{II})\right\}$.

Proof.

Construct a K-way disjunction of $\mathcal{S}$ with
$\mathcal{S}_k := \{(y, T, z) : yM^k = z, T = M^k\}$. □

Linearization Step: Final Formulation

Apply the proposition with $\mathcal{Y} = \mathcal{U}_{n-1}$, $y = u_{n-1}$, $T = T_n$ and $z = u_n$.

Linearization Step: Final Formulation

Apply the proposition with $\mathcal{Y} = \mathcal{U}_{n-1}$, $y = u_{n-1}$, $T = T_n$ and $z = u_n$.

$$\max_{u,v,\mu} f(u_N)$$

s.t. $u_0 = p$

$$\sum_{k=1}^{K} v_{n-1,k} = u_{n-1} \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} v_{n-1,k} M^k = u_n \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} \mu_{n-1,k} = 1 \qquad n = 1, \ldots, N$$

$$v_{n-1,k} \in \mathcal{U}_{n-1}\mu_{n-1,k},\ \mu_{n-1,k} \in \{0,1\} \qquad n = 1, \ldots, N, k = 1, \ldots, K.$$

Linearization Step: Final Formulation

Apply the proposition with $\mathcal{Y} = \mathcal{U}_{n-1}$, $y = u_{n-1}$, $T = T_n$ and $z = u_n$.

$$\max_{u,v,\mu} f(u_N)$$

$$\text{s.t. } u_0 = p$$

$$\sum_{k=1}^{K} v_{n-1,k} = u_{n-1} \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} v_{n-1,k} M^k = u_n \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} \mu_{n-1,k} = 1 \qquad n = 1, \ldots, N$$

$$v_{n-1,k} \in \mathcal{U}_{n-1}\mu_{n-1,k},\ \mu_{n-1,k} \in \{0,1\} \qquad n = 1, \ldots, N, k = 1, \ldots, K.$$

- The relation $v_{n-1,k} \in \mathcal{U}_{n-1}\mu_{n-1,k}$ serves as a "big-M constraint". Any set that outer-approximates the polytope $\mathcal{U}_n$ can also be used.
- For each specific application, we will provide such reasonable sets.

Optimization Model

$$\max_{u,v,\mu} \; f(u_N)$$

$$\text{s.t. } u_0 = p$$

$$\sum_{k=1}^{K} v_{n-1,k} = u_{n-1} \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} v_{n-1,k} M^k = u_n \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} \mu_{n-1,k} = 1 \qquad n = 1, \ldots, N$$

$$v_{n-1,k} \in \mathcal{U}_{n-1}\mu_{n-1,k}, \;\; \mu_{n-1,k} \in \{0,1\} \qquad n = 1, \ldots, N, k = 1, \ldots, K.$$

Optimization Model

$$\max_{u,v,\mu} \; f(u_N)$$

$$\text{s.t. } u_0 = p$$

$$\sum_{k=1}^{K} v_{n-1,k} = u_{n-1} \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} v_{n-1,k} M^k = u_n \qquad n = 1, \ldots, N$$

$$\sum_{k=1}^{K} \mu_{n-1,k} = 1 \qquad n = 1, \ldots, N$$

$$v_{n-1,k} \in \mathcal{U}_{n-1}\mu_{n-1,k}, \;\; \mu_{n-1,k} \in \{0,1\} \qquad n = 1, \ldots, N, k = 1, \ldots, K.$$

- We have K drugs, each with a transition probability matrix $M^k \in \mathbb{R}^{d \times d}$, (with $d = 2^g$).
- The size of the auxiliary matrices is $1 \times d$ ($r = 1$).
- Matrix p is the initial distribution.
- We have $\mathcal{U}_n = \Delta_d := \left\{x \in \mathbb{R}^d_+ : \; \sum_{j=1}^{d} x_j = 1\right\}$.
- $f(w) := wq^T$ is linear, where q is the desired final distribution.

Results–A Real Dataset with $K = 15$ Drugs (Mira et al., 2015)

initial	$N = 4$	$N = 5$	$N = 6$	$N = 7$	$N = 8$	$N = 9$	$N = 10$
1000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
0100	0.375	0.458	0.458	0.463	0.463	0.471	0.479
0010	0.500	0.500	0.500	0.512	0.512	0.515	0.516
0001	0.667	0.667	0.667	0.690	0.690	0.693	0.693
1100	0.389	0.389	0.458	0.458	0.463	0.463	0.471
1010	0.583	0.583	0.587	0.587	0.591	0.591	0.596
1001	0.667	0.667	0.690	0.690	0.693	0.693	0.696
0110	0.333	0.375	0.458	0.458	0.463	0.463	0.471
0101	0.458	0.458	0.463	0.463	0.471	0.479	0.479
0011	0.500	0.500	0.500	0.502	0.531	0.539	0.553
1110	0.333	0.333	0.375	0.458	0.458	0.463	0.463
1101	0.375	0.458	0.458	0.463	0.463	0.471	0.479
1011	0.333	0.389	0.417	0.458	0.458	0.475	0.475
0111	0.198	0.333	0.375	0.458	0.458	0.463	0.463
1111	0.333	0.375	0.458	0.458	0.463	0.463	0.471
MILP (sec)	0.15	0.25	0.34	0.55	0.95	1.99	4.98
BBNode	9.07	57.00	154.00	378.00	972.67	2423.47	6733.47
Enum (sec)	0.94	15.90	273.02	$4.1 \cdot 10^3$	$6.1 \cdot 10^4$	$9.2 \cdot 10^5$	$1.4 \cdot 10^7$

- Probabilities of going from initial state to wild type (0000) are reported.
- MILP (sec): Avg. CPU time of Gurobi (absolute opt. gap is set to 0.001).
- Enum (sec): Avg. (serial) time of the enumeration (estimated for $N > 6$).

Results–Randomly Generated Instances

- We "synthetically generated" K drugs with different string sizes g.
- Average MILP run times are reported (in seconds).

g	d	K	$N=5$	$N=6$	$N=7$	$N=8$	$N=9$	$N=10$
4	16	15	0.26	0.45	0.61	1.32	2.99	6.05
4	16	20	0.42	0.58	1.04	1.91	4.25	8.96
4	16	25	0.53	0.73	1.06	2.14	4.40	8.49
4	16	30	0.56	0.85	1.48	3.03	6.69	16.55
5	32	15	0.46	0.77	1.72	3.99	10.95	33.32
5	32	20	0.68	1.23	2.92	8.12	23.54	109.14
5	32	25	0.86	1.80	4.41	12.54	44.90	233.08
5	32	30	1.03	2.29	5.44	17.97	82.17	439.48

Results–Randomly Generated Instances

- We "synthetically generated" K drugs with different string sizes g.
- Average MILP run times are reported (in seconds).

g	d	K	$N=5$	$N=6$	$N=7$	$N=8$	$N=9$	$N=10$
4	16	15	0.26	0.45	0.61	1.32	2.99	6.05
4	16	20	0.42	0.58	1.04	1.91	4.25	8.96
4	16	25	0.53	0.73	1.06	2.14	4.40	8.49
4	16	30	0.56	0.85	1.48	3.03	6.69	16.55
5	32	15	0.46	0.77	1.72	3.99	10.95	33.32
5	32	20	0.68	1.23	2.92	8.12	23.54	109.14
5	32	25	0.86	1.80	4.41	12.54	44.90	233.08
5	32	30	1.03	2.29	5.44	17.97	82.17	439.48

- We have $O(NK)$ matrix variables, each of size $r \times d$ ($r = 1$), and $O(NK)$ binary variables.