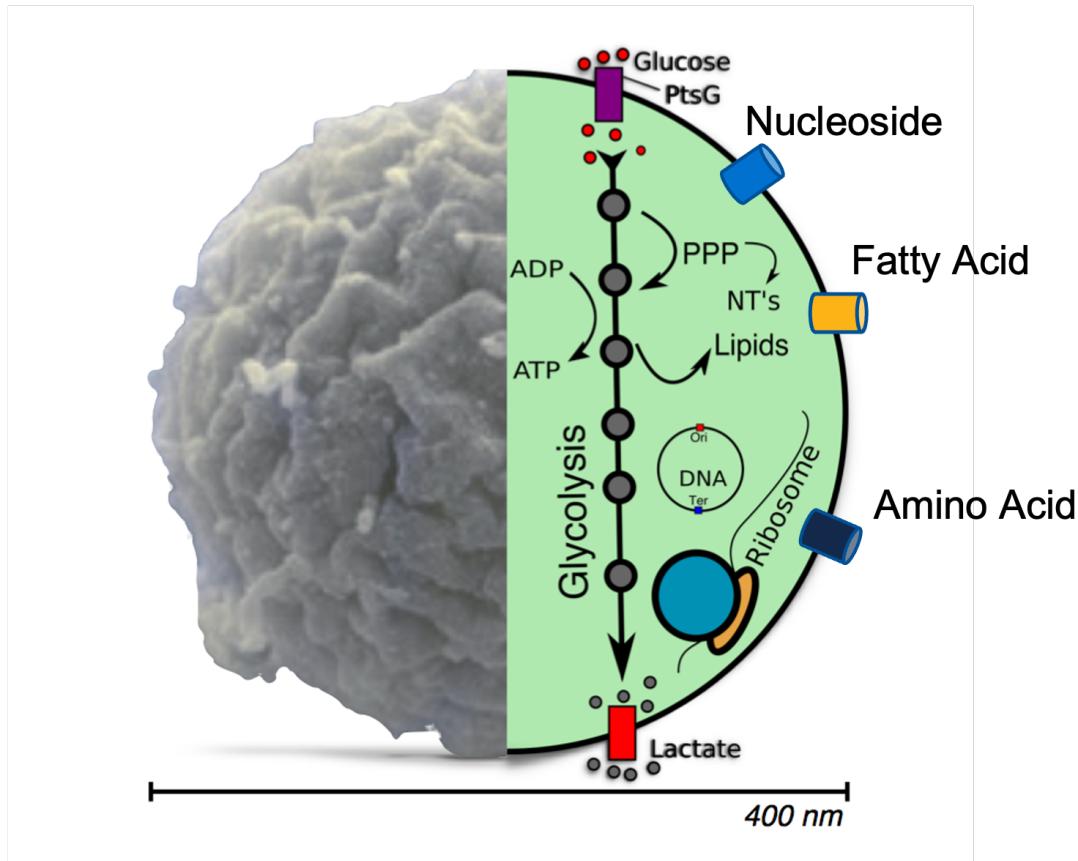


Coupled Genetic Information Processes and Metabolism in Minimal Cell, JCVI-syn3A



STC-QCB Summer School
August 5-9, 2024

Enguang Fu and Zaida Luthey-Schulten

University of Illinois at Urbana-Champaign

Description

This Tutorial describe how to use the software to perform and analyze hybrid stochastic/deterministic simulations of a minimal bacterium.

Lattice Microbes development is supported in part by NSF Science and Technology Center for Quantitative Cell Biology (NSF DBI-2243257) and NSF (MCB: 2221237 “Simulating a growing minimal cell: Integrating experiment and theory”).

Contents

List of Figures	iv
List of Tables	v
Chapter 1 Introduction	1
1.1 Lattice Microbe Overview	1
1.2 pyLM Overview	1
1.3 jLM Overview	3
1.4 Stochastic Modeling	5
Chapter 2 Tutorials for CME in pyLM	6
2.1 Tutorial 1: Bimolecular Reaction Solved in ODE and CME	6
2.2 Tutorial 2: Genetic Information Process Model in CME	11
2.3 Tutorial 3: CME/ODE Whole Cell Model of Minimal Cell, JCVI-syn3A	15
2.3.1 Minimal Cell, JCVI-syn3A and its Minimal Genome	15
2.3.2 Genetic Information Processes and Metabolism of JCVI-syn3A	16
2.3.3 Hybrid CME/ODE Algorithm	21
2.3.4 Implementation in Coding	22
2.3.5 Results Analysis	23
Chapter 3 License and Copyright	28
Bibliography	29

List of Figures

1.1	A schematic Architecture of the Lattice Microbes software.	2
1.2	Workflow in pyLM and jLM. In jLM, regions and diffusion rules are defined. In the hookSimulation(), various algorithms can be incorporated with CME or RDME. In this tutorial, we will show hybrid CME/ODE and RDME/ODE algorithm implemented by Lattice Microbe	4
1.3	Lattice Microbe supports stochastic CME and RDME. With hookSimulation, CME and RDME can communicate with ODE (for metabolism), BD (for chromosome model) and PDE.	5
2.1	Contrast Between Deterministic and Stochastic Solutions to Bimolecular Reaction. Left: Deterministic solution to the bimolecular reaction. Right: One representative stochastic solution of the bimolecular reaction.	9
2.2	Left: Ensemble Average and Variance of 10 replicates in Stochastic Bimolecular Reaction. Right: Ensemble Average and Variance of 100 replicates in Stochastic Bimolecular Reaction.	10
2.3	Genetic Information Processing Reaction Systems and its Chemical Master Equation	11
2.4	Single Cell Trace of mRNA and protein. Left: Stair-stepping trace of mRNA. Right: Burst of protein due to the stochastic mRNA's translation.	13
2.5	Average and Variance of mRNA and protein among 10 replicates. Left: Ensemble Averaged mRNA counts and its variance over 6300 seconds. Right: Ensemble Averaged protein counts and its variance over 6300 seconds.	14
2.6	Distributions of protein counts. Left: Protein counts of 100 replicate at the end of the cell cycle. Right: Analytical Solution at steady state.	14
2.7	Left: Protein Coding Genes of JCVI-syn3A. Only less than 90 genes are unclear. Right: JCVI-syn3A with genetic information processes and metabolism shown . .	15
2.8	Left: Genetic Information Flow. Right: Functions of Seven Genetic Information Processes	16
2.9	Left: oriC region. 9 nucleotide signature binding with DnaA domain IV shown in yellow and red, 3 nucleotide AT rich region binding with DnaA domain III shown in grey Right: PDB structure of DnaA domain IV and domain III binding with chromosome a): [1] b): [2]	16
2.10	Three stage DNA replication initiation	17
2.11	Ordered and Bidirectional Replication	17
2.12	Supply of GTP and dGTP for DNA and RNA synthesis in nucleotide metabolism. The metabolites in the bracket come from central metabolism	19
2.13	Whole Metabolism of JCVI-syn3A with five subsystems: central, nucleotide, lipid, cofactor and amino acid	19

2.14	Central Metabolism in JCVI-syn3A. Phosphoenolpyruvate (pep) is both the upstream and downstream of glycolysis. 1,3-Bisphosphoglyceric acid (1,3dpg), Phosphoribosyl pyrophosphate (prpp) and pep are in nucleotide metabolism.	20
2.15	Mechanism of reaction A+X to B+Y in random binding model. [E] is the concentration of enzyme, [A] concentration of molecule A	21
2.16	Communication Between CME and ODE to simulate the co-evolution of Genetic Information Processes and Metabolism.	22
2.17	Traces of pep concentration. Left:Pep concentration goes to zero for one unhealthy/dead cell. Right:Pep concentration of 10 healthy replicates.	24
2.18	Cell surface area and volume increase.	24
2.19	Distributions of rounds of initiation and the one possible theta structure of circular chromosomes of 5 rounds of replication initiation.	25
2.20	Distributions of protein counts. Left:Protein counts of 100 replicate at the end of the cell cycle. Right:Analytical Solution at steady state.	25
2.21	Gene Copy Numbers and the Distribution of Copy number at the end of the cell cycle.	26
2.22	Traces of free mRNA, R_0001 of 10 replicates and the number of all mRNAs over the cell cycle.	26
2.23	Ensemble averaged count of different proteins over the cell cycle and the distribution of scaled protein abundances.	27
2.24	Traces of Four Metabolites Over the Entire Cell Cycle	27

List of Tables

2.1	Reactions and Units in both CME and RDME. Here, the stochastic rate constant should be computed from the macroscopic rate constant using the volume of the experiment, V , and Avogadro's number, N_A	7
2.2	Four reactions with their rate constants.	11
2.3	Reactions and Rates for Replication, Transcription, Translation and Degradation . .	18
2.4	Rate Form for Replication, Transcription, Translation and Degradation	18
2.5	Concentrations of Species in Genetic Information Processes and Metabolism . . .	22
2.6	Input Files and Information Read In	23

Chapter 1

Introduction

1.1 Lattice Microbe Overview

Lattice microbe (LM) is a GPU accelerated stochastic simulation platform. The latest published version 2.4 is in https://github.com/Luthey-Schulten-Lab/Lattice_Microbes. In this tutorial, we use an updated version 2.5 under development.

LM is designed to simulate stochastic processes in biological cells using the Chemical Master Equation (CME) and the Reaction Diffusion Master Equation (RDME). With hooking, one can also incorporate metabolic reactions using Ordinary Differential Equation (ODE) solver and chromosome dynamics with Brownian Dynamics (BD) code.

The architecture of LM is shown in Figure 1.1. pyLM and jLM are two python-based Problem Solving Environments (PSE) developed in 2013 [3] and 2018, respectively. The API of pyLM and jLM can be found at <https://luthey-schulten.chemistry.illinois.edu/software/LM2.4/API.html>.

With pyLM and jLM, one can easily construct the system purely using Python. And the calculation is still performed in C++.

1.2 pyLM Overview

pyLM is a Problem Solving Environment (PSE) for biological simulations [3]. Written in Python, it wraps and extends Lattice Microbes. The PSE is comprised of a base set of functionality to set up, monitor and modify simulations, as well as a set of standard post-processing routines that interface to other Python packages, including NumPy, SciPy, H5py, iGraph to name a few.

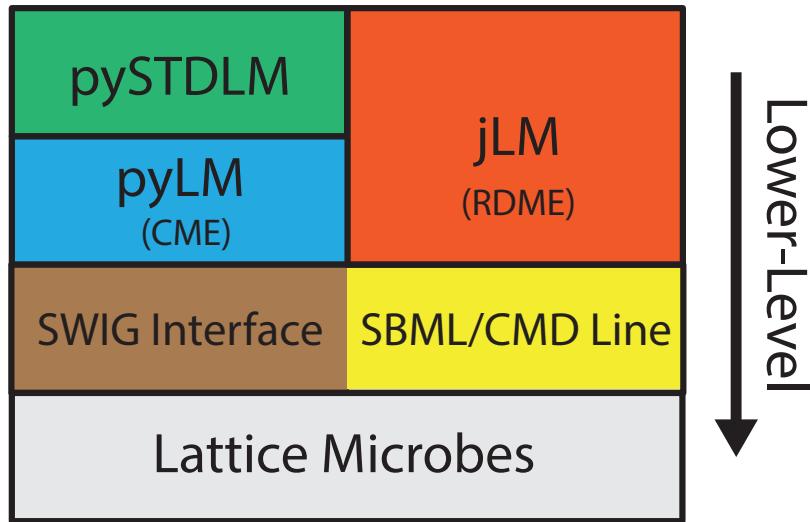


Figure 1.1: A schematic Architecture of the Lattice Microbes software.

The PSE is shown schematically in Figure 1.1. It sits on top of a SWIG interface that allows the C++ code to be accessible from the Python terminal. Using pyLM allows the user to set up, run and post-process simulations all within a single script. A general workflow is shown in Figure 1.2. For tutorials on using pyLM please see the “pyLM Reference pdf” in the Luthey-Schulten webpage under Software/Documentations.

Capabilities of pyLM and pySTDLM

pyLM and the included library of standard systems pySTDLM provide a problem solving environment for setting up, running and analyzing stochastic biological simulations [3]. It contains functionality for specifying simulation setup including:

- Named species
- Initial counts and distributions
- Reactions and rates
- Spatial localization and definition
- Diffusion properties
- Obstacles
- Handles to data in the popular Numpy array representation
- Plotting species averages/variances and individual time traces
- Plotting Kymographs of spatial species distributions
- Reaction network generation
- Dynamic reaction network representations

1.3 jLM Overview

jLM is also written in python and designed as an interface for Jupyter Notebook and JupyterLab for RDME simulation. It offers a suite of functions to visualize the RDME simulation states like spatial regions, species and reactions.

Capabilities of jLM

jLM contains a set of different methods and functions to build up a RDME or RDME hybrid simulation. It contains functionality for specifying simulation setup including:

- Named species
- Initial counts and distributions
- Reactions and rates
- Spatial localization and definition
- Diffusion properties
- Visualization of the RDME simulation spatial geometry, reactions and species

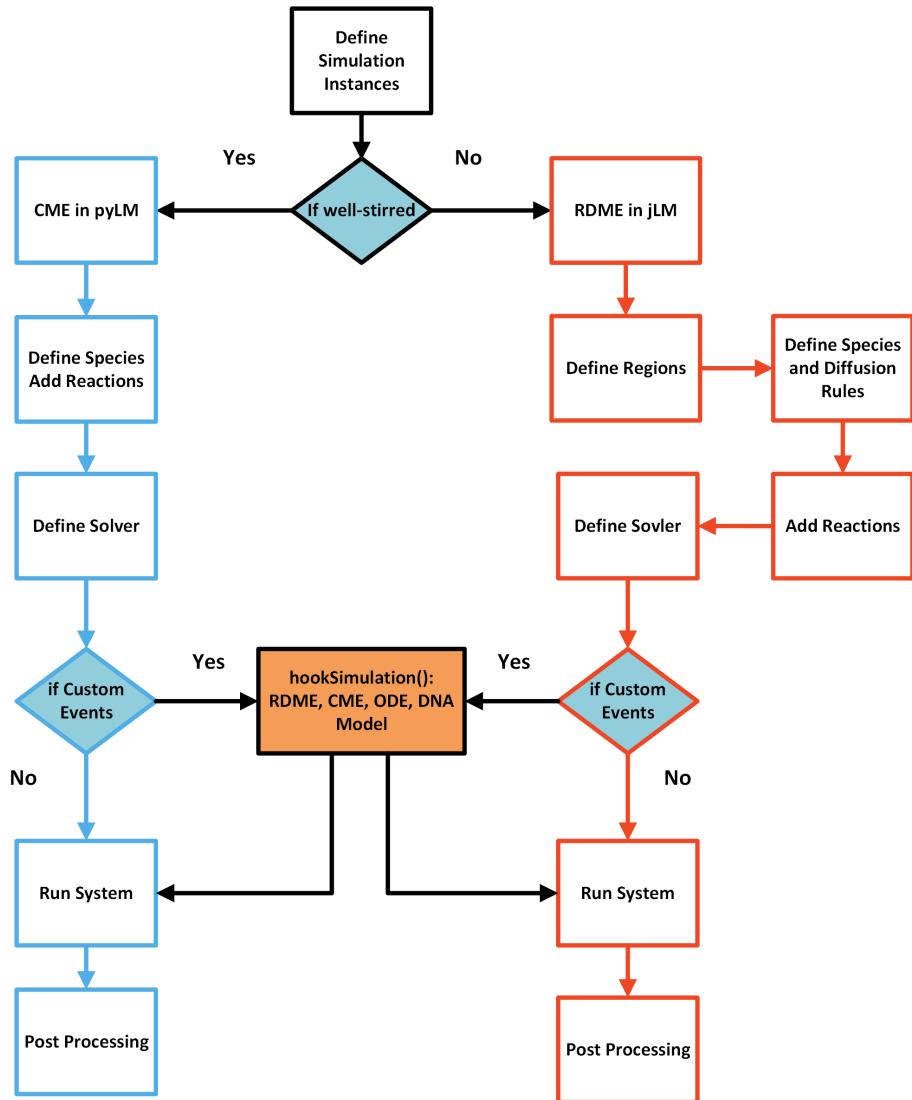


Figure 1.2: Workflow in pyLM and jLM. In jLM, regions and diffusion rules are defined. In the hookSimulation(), various algorithms can be incorporated with CME or RDME. In this tutorial, we will show hybrid CME/ODE and RDME/ODE algorithm implemented by Lattice Microbe

1.4 Stochastic Modeling

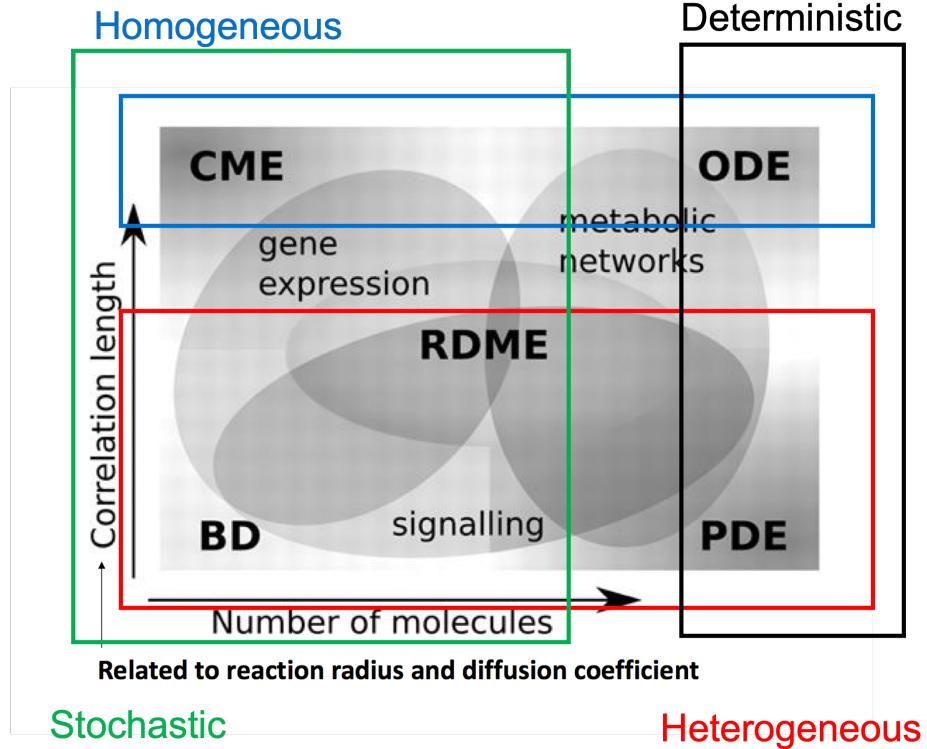


Figure 1.3: Lattice Microbe supports stochastic CME and RDME. With hookSimulation, CME and RDME can communicate with ODE (for metabolism), BD (for chromosome model) and PDE.

Lattice Microbes can be used to simulate chemical master equations (CME):

$$\frac{dP(\mathbf{x}, t)}{dt} = \sum_r^R [-a_r(\mathbf{x})P(\mathbf{x}, t) + a_r(\mathbf{x}_\nu - \mathbf{S}_r)P(\mathbf{x} - \mathbf{S}_r, t)]$$

and reaction-diffusion master equations (RDME):

$$\begin{aligned} \frac{dP(\mathbf{x}, t)}{dt} &= \sum_\nu^V \sum_r^R [-a_r(\mathbf{x}_\nu)P(\mathbf{x}_\nu, t) + a_r(\mathbf{x}_\nu - \mathbf{S}_r)P(\mathbf{x}_\nu - \mathbf{S}_r, t)] \\ &+ \sum_\nu^V \sum_\xi^{\pm\hat{i}, \hat{j}, \hat{k}} \sum_\alpha^N [-d_{xi}^\alpha x_\nu^\alpha P(\mathbf{x}, t) + d_\xi^\alpha (x_{\nu+\xi}^\alpha + 1_\nu^\alpha)P(\mathbf{x} + 1_{\nu+\xi}^\alpha - 1_\nu^\alpha, t)] \end{aligned}$$

RDME is used to simulate spatially heterogeneous systems. Within each subvolume, the reactions are assumed to be well-stirred and described by CME.

Chapter 2

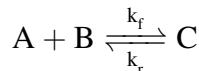
Tutorials for CME in pyLM

2.1 Tutorial 1: Bimolecular Reaction Solved in ODE and CME

In the first tutorial, we will motivate stochastic modeling with a simple example of a bimolecular reaction. The counts of reactants are small enough that you will see the fluctuations and variance of the stochastic result compared to the deterministic one.

You will use **Tut1.1-ODEBimol.ipynb** to run the ODE of Bimolecular reaction solved with Scipy and **Tut1.2-CMEBimol.ipynb** to run the CME of the Bimolecular reaction with pyLM in Lattice Microbe.

We will simulate the association/dissociation reaction of hypothetical molecules:



Let us start out with a low number of each particles: 100 of A and B and 0 of C. Let us imagine simulating that problem in a microbe sized volume of 1 fL . Also, let us start off with rates of $k_f = 1.07 \times 10^6 M^{-1}s^{-1}$ and $k_r = 0.351/s$.

Under ODE representation using concentrations of species, the equations are:

$$\begin{aligned}\frac{d[A]}{dt} &= -k_f[A][B] + k_r[C] \\ \frac{d[B]}{dt} &= -k_f[A][B] + k_r[C] \\ \frac{d[C]}{dt} &= k_f[A][B] - k_r[C]\end{aligned}$$

Since we are using counts rather than concentrations in CME, the forward rate constant (2nd order) is divided by Avogadro's number and the volume of the cell which takes us to units of molecules per second. Conversion to stochastic rate constant is shown in Table 2.1.

Table 2.1: Reactions and Units in both CME and RDME. Here, the stochastic rate constant should be computed from the macroscopic rate constant using the volume of the experiment, V , and Avogadro's number, N_A .

Order	Form	Parameters	Macroscopic Units	Stochastic Rate Constant (s^{-1})
0th	$\emptyset \rightarrow A$	k_{0th}	$M \cdot s^{-1}$	$k_{0th} \cdot V \cdot N_A$
1st	$A \rightarrow B$	k_{1st}	s^{-1}	k_{1st}
2nd	$A + B \rightarrow C$	k_{2nd}	$M^{-1} \cdot s^{-1}$	$\frac{k_{2nd}}{V \cdot N_A}$
2nd (Self)	$2A \rightarrow B$	k_{2nd}	$M^{-1} \cdot s^{-1}$	$\frac{k_{2nd}}{V \cdot N_A}$

Now go to Jupyter Notebook file **Tut1.1-ODEBimol.ipynb** and **Tut1.2-CMEBimol.ipynb** (latter's code shown in List 2.1).

Listing 2.1: Tut1.2-CMEBimol — Code used to solve the bimolecular reaction with the discrete/stochastic CME implementation using Lattice Microbes.

```

1 # Import Standard Python Libraries
2 import os
3 import numpy as np
4 import matplotlib.pyplot as plt
5
6 # Import pyLM Libraries
7 from pyLM import *
8 from pyLM.units import *
9 from pySTDLML import *
10 from pySTDLML.PostProcessing import *
11
12 # Constants
13 V = 1.0e-15      # L
14 NA = 6.022e23    # molecules/mole
15 kf_ODE = 1.07e6; # /M/s
16 kr_ODE = 0.351; # /s
17 kf = 1.07e6/(NA*V) # convert from 1.07e5 /M/s to /counts/s
18 kr = 0.351        # /s
19
20 ### Define a CME Simulation
21
22 # Create our CME simulation object
23 sim = CME.CMESimulation(name='Bimolecule Reaction')
24
25 # Define our chemical species
26 species = ['A', 'B', 'C']
27 sim.defineSpecies(species)
28
29 # Add reactions to the simulation
30 sim.addReaction(reactant=['A', 'B'], product='C', rate=kf)
31 sim.addReaction(reactant='C', product=['A', 'B'], rate=kr)
32
33 # Set our initial species counts
34 sim.addParticles(species='A', count=50)

```

```

35 sim.addParticles(species='B', count=50)
36 sim.addParticles(species='C', count=0)
37
38 # Define simulation parameters: run for 30 seconds, saving data every 30 ms
39 sim.setWriteInterval(microsecond(30))
40 sim.setSimulationTime(30)
41 filename = "./T1.2-bimol.lm"
42 os.system("rm -rf %s"%(filename))
43 sim.save(filename)
44
45 # Run reps replicates using the Gillespie solver
46 reps = 10
47 sim.run(filename=filename, method="lm::cme::GillespieDSolver", replicates=reps
        )
48
49 ### Post-Process Simulation
50 plotfolder = './plots_bimolecule/'
51
52 if not os.path.exists(plotfolder):
53     os.mkdir(plotfolder)
54
55 # Plot the traces of one replicate to see the fluctuation
56 fieHandle = PostProcessing.openLMFile(filename)
57 rep = 1
58 picturepath = plotfolder + 'bimolecule_CME_trace_replicate{0}.png'.format(rep)
59 PostProcessing.plotTrace(fieHandle, species=['A', 'C'], replicate=1, filename=
        picturepath)
60 closeLMFile(fieHandle)
61
62 # Plot average and variance among all replicates
63 picturepath = plotfolder + 'bimolecule_CME_Avgs_Vars_{0}replicates.png'.format(
        (reps))
64 PostProcessing.plotAvgVarFromFile(filename = filename, species = ['A', 'C'],
        outfile = picturepath)

```

As mentioned in the introduction part, pyLM Problem Solving Environment is intensively used to construct the CME system. To do so, we need to import pyLM module.

The line `sim=CME.CMESimulation()` creates an empty simulation object. The next lines define the chemical species; in pyLM species are named by python strings and must be registered with the simulation using the `defineSpecies` command. `addReaction` function adds reactions to the defined species.

In Lattice Microbes, you need to specify both the forward and back reactions separately. The first and second argument can be either a tuple of reactants or a string when only one reactant is specified. Lattice Microbes currently supports 0th, 1st and 2nd order reactions, and reaction rates must be specified in the “stochastic” format (see Table 2.1). In the special case of a 0th order reaction, the empty string ‘’ should be passed as the reactant. In addition, annihilation reactions can be specified by passing the empty string ‘’ as the product parameter.

The following lines with `addParticles` define the initial species counts.

Next the simulation parameters are specified, time steps will be written out every 30 microseconds and the total simulation will run for 30 seconds. The next line is of particular importance; the simulation must be saved to a file before running the simulation.

Finally, we call the `run(...)` command on the simulation object giving it the name of the simulation file, the simulation method and the number of independent trajectories (replicates) to run of that simulation. In this tutorial, we use Direct Gillespie Algorithm to sample stochastic bimolecular reaction system.

By running the simulation, you will see a long standard output showing the number of finished replicates. Generally, CME simulation will finish rather quickly.

`pySTDLML` is a library of standard functionality such as standard reaction systems, cell systems. In addition it contains a number of pre- and post-processing functionality. In the post processing part, we showed two build-in functions in `pySTDLML`: `PostProcessing.plotTrace` for a single replicate and `PostProcessing.plotAvgVarFromFile` for the whole ensemble to visualize the results.

Figure 2.1 shows the deterministic result from ODE and stochastic result from CME.

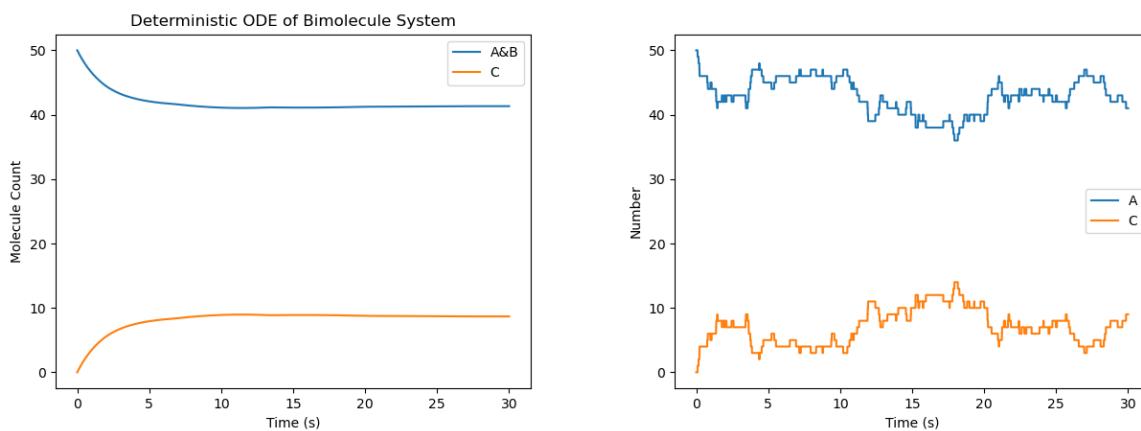


Figure 2.1: Contrast Between Deterministic and Stochastic Solutions to Bimolecular Reaction. Left: Deterministic solution to the bimolecular reaction. Right: One representative stochastic solution of the bimolecular reaction.

In ODE result, what you should note is that the count of each species varies smoothly across the time course. However, the count of each reactant must be in integer numbers due to the discreteness of molecules. Further, the reactions occur via the collision between two molecules, so the change of count also in integer units. Under such reasoning, the ODE result is not accurate anymore for microscopic reactions where the counts of reactants are low. Stochastic modeling was designed to address this point.

You might note that the behavior in stochastic result is qualitatively the same to the deterministic ODE result, however there appears to be considerable fluctuation, even after the system has come to equilibrium. This is due to the stochastic nature of the process, where the reaction can transiently fluctuate away from the equilibrium value. In addition, you may be able to tell that the changes in particle number from one time to another are in integer increments, though this will become considerably more obvious at lower number of particles.

The averaged and variance of species' count across the whole replicates are shown in Figure 2.2. The left one is the result of ten replicates and the right one for 100.

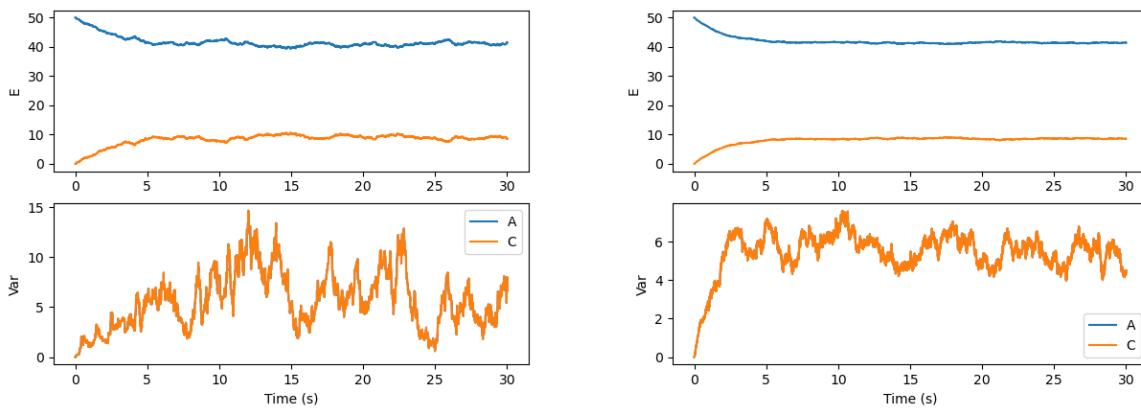


Figure 2.2: Left: Ensemble Average and Variance of 10 replicates in Stochastic Bimolecular Reaction. Right: Ensemble Average and Variance of 100 replicates in Stochastic Bimolecular Reaction.

Based on the preceding discussion, you are encouraged to explore the following questions.

Questions:

1. Try plotting the average and variance of different replicates by changing the number of replicates from 10 to 100 or even more. You need to restart the Jupyter Notebook kernal to start a new CME simulation. See Figure 2.2 for the situation with 10 and 100 replicates.
2. How many replicates are required to get a smooth average? How many for a smooth variance?
3. (Challenge) Can you derive an analytical solution for the system of equations? Try fitting the rate constants using the results of the stochastic simulations. You may find `scipy.optimize.curve_fit()` useful for this.
4. (Challenge) Does the theoretical average number from CME always be consistent/same with the count from ODE? You may find Page 420 in McQuarrie's classic paper STOCHASTIC APPROACH TO CHEMICAL KINETICS helpful.

2.2 Tutorial 2: Genetic Information Process Model in CME

In Tutorial 1, we discussed the influence of particles counts on fluctuations. Open ChatGPT and ask this question: Give an example on large differences in rate constants that lead to significant fluctuations. Also open this answer gene expression, which is our next tutorial.

Now go to Jupyter Notebook file **Tut2.1-GeneticInformationProcess.ipynb**. Here we will simulate a toy genetic information processing (GIP) model.

With 3 species and 4 reactions, the genetic information process starts from the transcription of gene to mRNA. mRNA can be translated to protein or degraded to its monomers. Protein can also be degraded. The reactions and rate constant is shown in Figure 2.3 and Table 2.2.

The rate constants are for DnaA Coding Gene (G_0001) of minimal cell. The first three rate constants are calculate based on the initial concentrations of nucleotides and amino acids charged tRNA in the Cell paper [4]. We also fix the gene copy number to 1 and assume initial count of mRNA to be 1. The initial count of protein is 0. The degradation rate of protein is estimated based half-life 25 hours [5].

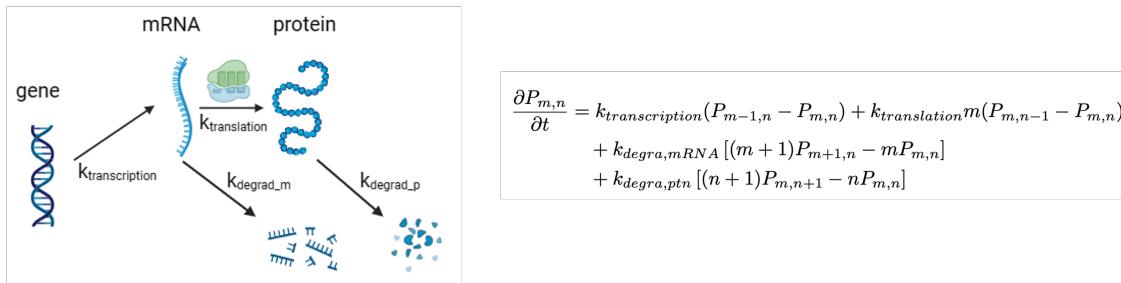


Figure 2.3: Genetic Information Processing Reaction Systems and its Chemical Master Equation

Table 2.2: Four reactions with their rate constants.

Names	Reaction	Rate Constant (s^{-1})	Propensity (s^{-1})
Transcription	Gene \rightarrow mRNA	$k_{transcription} = 6.41E - 4$	$k_{transcription}$
Degradation of mRNA	mRNA \rightarrow \emptyset	$k_{deg,m} = 2.59E - 3$	$k_{deg,m} N_{mRNA}$
Translation	mRNA \rightarrow mRNA + Protein	$k_{translation} = 7.20E - 2$	$k_{translation} N_{mRNA}$
Degradation of Protein	Protein \rightarrow \emptyset	$k_{deg,p} = 7.70E - 6$	$k_{deg,p} N_{ptn}$

In this given Jupyter Notebook, the default simulation length is **6300 seconds**, the entire cell cycle with write interval 1 second for **10 replicates**.

Listing 2.2: Tut .2 .1-GeneticInformationProcess — Code to set up and execute the gene expression model with CME.

```

1 # Import Standard Python Libraries
2 import os
3 import numpy as np
4 import matplotlib.pyplot as plt
5

```

```

6 # Import pyLM Libraries
7 from pyLM import *
8 from pyLM.units import *
9 from pySTDLML import *
10 from pySTDLML.PostProcessing import *
11
12 # Constants
13 k_transcription = 6.41e-4      # Transcription, s^-1
14 k_degra_mRNA = 2.59e-3       # degradation of mRNA, s^-1
15 k_translation = 7.2e-2        # translation, s^-1
16 k_degra_ptn = 7.70e-6        # degradation of protein, s^-1
17
18 ### Define CME simulation
19
20 # Create our CME simulation object
21 sim = CME.CMESimulation(name='Gene Expression')
22
23 # Define our chemical species
24 species = ['gene', 'mRNA', 'ptn']
25 sim.defineSpecies(species)
26
27 # Add reactions to the simulation
28 sim.addReaction(reactant='gene', product=('gene', 'mRNA'), rate=k_transcription)
29 sim.addReaction(reactant='mRNA', product='', rate=k_degra_mRNA)
30 sim.addReaction(reactant='mRNA', product=('mRNA', 'ptn'), rate=k_translation)
31 sim.addReaction(reactant='ptn', product='', rate=k_degra_ptn)
32
33 sim.addParticles(species='gene', count=1)
34 sim.addParticles(species='mRNA', count=1)
35 sim.addParticles(species='ptn', count=0)
36
37 # Simulation time is 6300 (entire cell life cycle of Minimal Cell).
38 writeInterval = 1
39 simtime = 6300
40
41 sim.setWriteInterval(writeInterval)
42 sim.setSimulationTime(simtime)
43
44 filename = "./T2.1-GeneticInformationProcess.lm"
45 os.system("rm -rf %s"%(filename)) # Remove previous LM file
46 sim.save(filename)
47
48 # Run multiple replicates using the Gillespie solver
49 reps = 100
50 sim.run(filename=filename, method="lm::cme::GillespieDSolver", replicates=reps)
51
52 ### Post-Processing
53 plotfolder = './plots_GeneticInformationProcess/'
54
55 if not os.path.exists(plotfolder):
56     os.mkdir(plotfolder)
57

```

```

58 # Plot the trace of mRNA or protein of a single replicate
59 rep = 3
60 picturepath = plotfolder + 'mRNA_replicate{0}.png'.format(rep)
61 PostProcessing.plotTraceFromFile(filename, species=['mRNA'], replicate=rep,
       outfile=picturepath )
62 picturepath = plotfolder + 'ptn_replicate{0}.png'.format(rep)
63 PostProcessing.plotTraceFromFile(filename, species=['ptn'], replicate=rep,
       outfile=picturepath )

```

First, we can plot the traces of mRNA and protein in a single replicate to see the stair-stepping trace of mRNA and the burst of protein in Figure 2.4. You are encouraged to plot different replicates to see the various patterns of protein burst over the entire cell cycle.

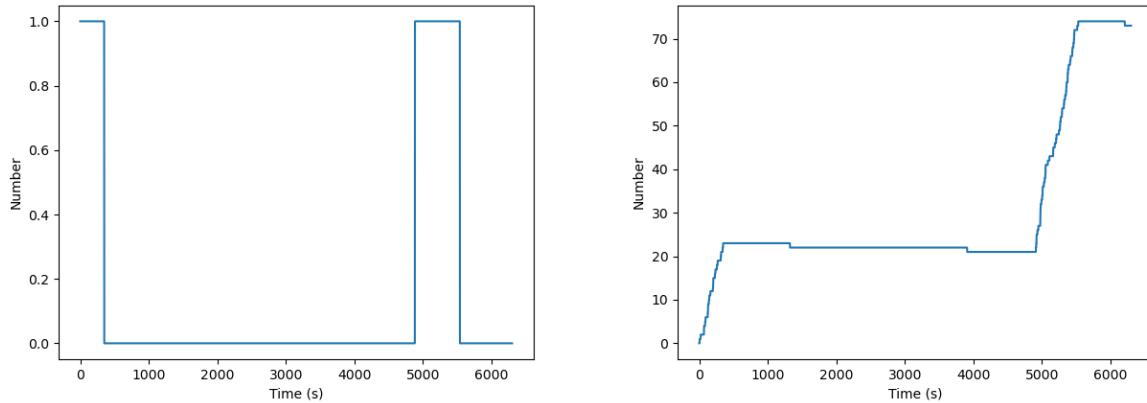


Figure 2.4: Single Cell Trace of mRNA and protein. Left: Stair-stepping trace of mRNA. Right: Burst of protein due to the stochastic mRNA's translation.

We can also see the average and variance of mRNA and protein counts over the whole simulation time span.

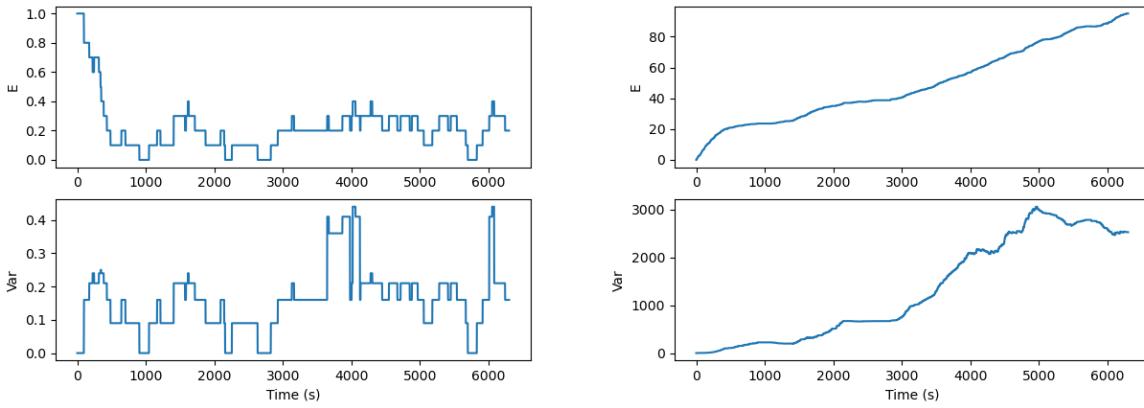


Figure 2.5: Average and Variance of mRNA and protein among 10 replicates. Left: Ensemble Averaged mRNA counts and its variance over 6300 seconds. Right: Ensemble Averaged protein counts and its variance over 6300 seconds.

Questions:

1. Do mRNA and protein reach steady-state during the 6300 seconds' simulation? How can you tell this from the plots?
2. The initial count of protein P_0001/DnaA from experimental proteomics data is 148 (Check Figure 2.6). Compare the mean count of protein at the end of the cell cycle to this experimental count. Does the simulation roughly generate 148 proteins during the entire cell cycle? And why this is important?

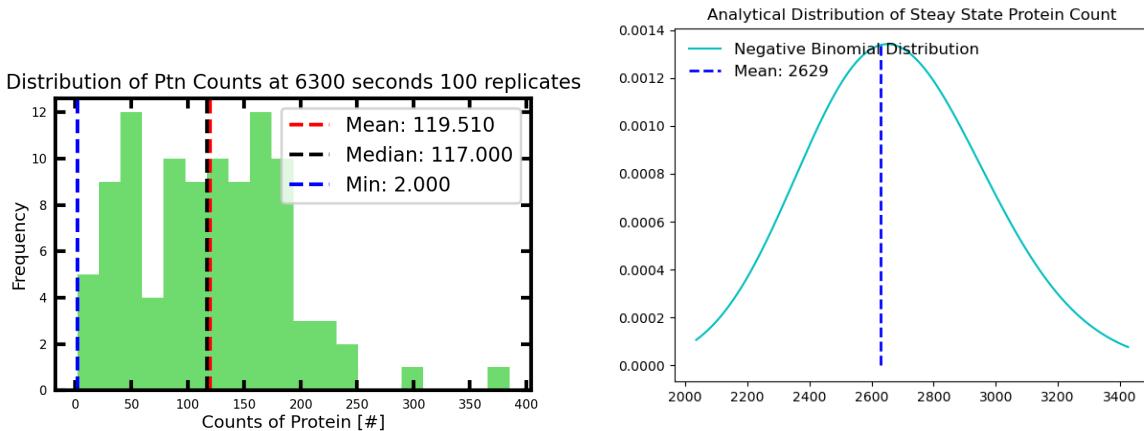


Figure 2.6: Distributions of protein counts. Left: Protein counts of 100 replicate at the end of the cell cycle. Right: Analytical Solution at steady state.

3. *Challenge* : What should be the distribution of protein counts among different replicates? Try to increase the replicates to 100 and plot the distribution. You can find an analytical solution under steady state in Swain's 2008 classic paper [6]. The distribution of steady state protein distribution is shown in Figure 2.6. Why the two distributions are so different? Does protein count reach steady state during the simulation?

2.3 Tutorial 3: CME/ODE Whole Cell Model of Minimal Cell, JCVI-syn3A

In this section, we will show you how to use Lattice Microbe to simulate the Minimal Cell with hybrid CME/ODE algorithm.

Before we moving on, please launch bash file **mpirun.sh** to simulate 4 cells for 2 minutes in parallel.

2.3.1 Minimal Cell, JCVI-syn3A and its Minimal Genome

Minimal cell, JCVI-syn3A is a synthetic bacterium based on mother organism Mycoplasma mycoides capri published by J. Craig Venter Institute in 2016 [7]. JCVI-syn3A has a doubling time of about 2 hours and consistently forms spherical cells of approximately 200 nm in radius.

JCVI-syn3A's genome is minimal in all living organism, 543 kbp long and containing 452 genes code for proteins and 38 genes for RNAs. The ratio of unclear protein-coding genes is 90 out of 452, which is also smallest compared to other well-studied organism, including yeast and E. coli [8].

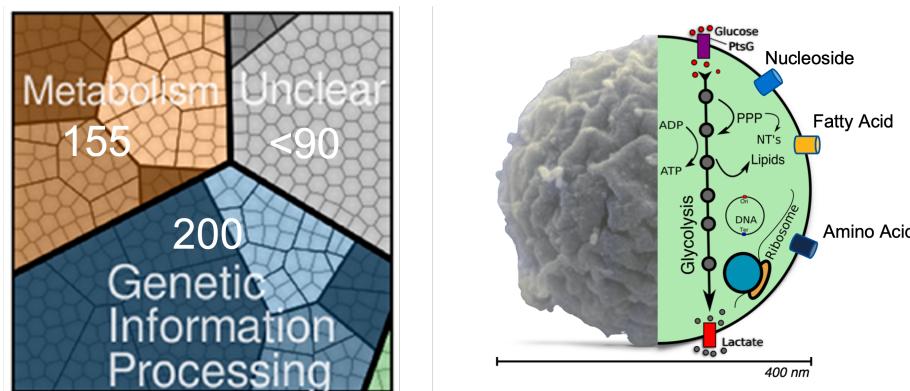


Figure 2.7: Left: Protein Coding Genes of JCVI-syn3A. Only less than 90 genes are unclear. Right: JCVI-syn3A with genetic information processes and metabolism shown

The genome information of JCVI-syn3A is stored as standard Genbank file on NCBI with ACCESSION Number CP016816.2. The Genbank file holds much more information than the pure nucleotide sequence.

Go to Genbank file under input_data folder and open with text editor. Search one example JCVISYN3A_0011. JCVISYN3A_0011 is the LocusTag of this certain gene in organism JCVI-syn3A, serving as a unique identifier. 0011 is the locusNum that will be heavily used in the modelling to distinguish genes, RNAs, and proteins from each other. The start and end index of JCVISYN3A_0011 is 15153 and 16799, respectively and the nucleotide sequence is at the end of the Genbank file. The function of gene JCVISYN3A_0011 is protein-coding gene, where the protein is "Nucleoside Transporter ABC substrate-binding protein" with amino acid sequence shown. This protein is one subunit of ribonucleoside (RNS) ABC import system that is assumed to import all nucleosides.

2.3.2 Genetic Information Processes and Metabolism of JCVI-syn3A

Genetic Information Processes in CME

Genetic information processes connect the blueprint genes to functional proteins. In our current whole cell model of minimal cell, we mainly consider seven types of processes listed in table. Replication copies the genetic information with the regulation of replication initiation. Transcription copies sequential information from DNA to RNA. There are three types of RNA, including mRNA, rRNA and tRNA that are tightly connected by translation. Translation takes place on ribosomes, where mRNA is read and an amino acid chain is generated according to the sequence of mRNA. rRNA, together with other ribosome proteins make up ribosomes in ribosomal biogenesis [9]. tRNA is the carriers and identifier of amino acid to the ribosome.

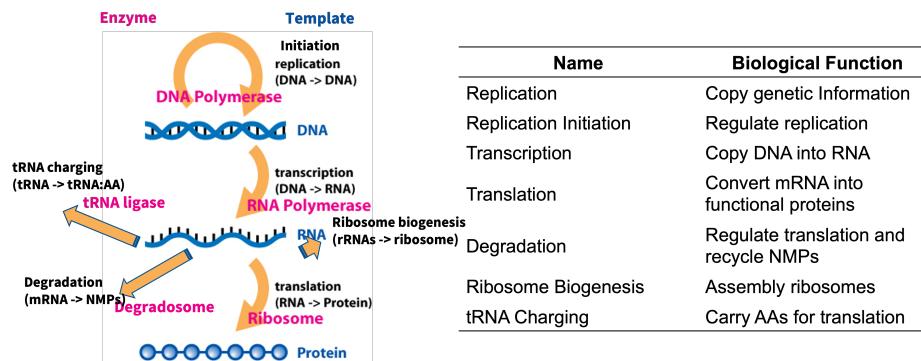


Figure 2.8: Left: Genetic Information Flow. Right: Functions of Seven Genetic Information Processes

As the busiest species in genetic information processes, mRNA can also be degraded by binding with degradosome. This competition of mRNA to bind with ribosome or degradosome is an important pathway to regulate genetic information processes that will be shown in the following result.

Replication initiation is modeled by the binding of multi-domain protein DnaA (encoded by JCVISYN3A_0001) and replisome with certain region called oriC on the chromosome [5].

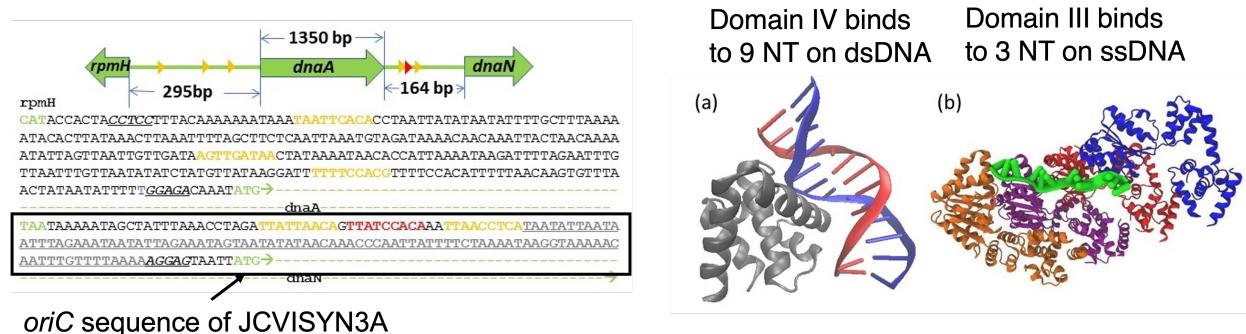


Figure 2.9: Left: oriC region. 9 nucleotide signature binding with DnaA domain IV shown in yellow and red, 3 nucleotide AT rich region binding with DnaA domain III shown in grey. Right: PDB structure of DnaA domain IV and domain III binding with chromosome a): [1] b): [2]

First stage of initiation is the binding of DnaA's Domain IV with nine-nucleotide signatures on double-strand DNA (three of them, shown in red and yellow in 2.9). This binding will open a pocket for DnaA's Domain III to bind with AT rich region on single-strand DNA following the nine-nucleotide signatures to build a filament of DnaA on single-strand DNA. The final step is the binding of replisome with the chromosome after the filament grows above 15 DnaA with 30 DnaA maximum due to the length of AT rich region is 90. In current model, the assembly of replisome is not explicitly included.

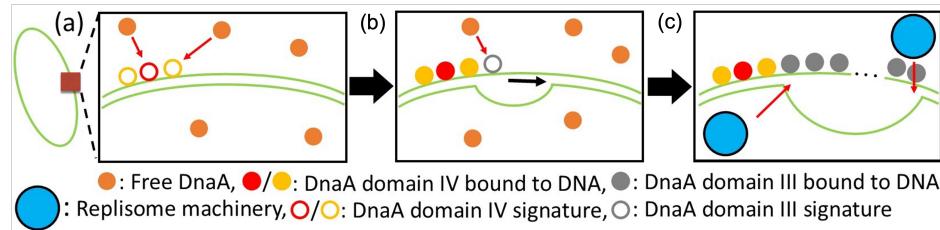


Figure 2.10: Three stage DNA replication initiation

Replication occurs gene by gene when replisomes move along the circular chromosome from oriC to ter in two directions. Since the locations of genes are fixed, the replication happens in order. So our replication model is ordered and bidirectional.

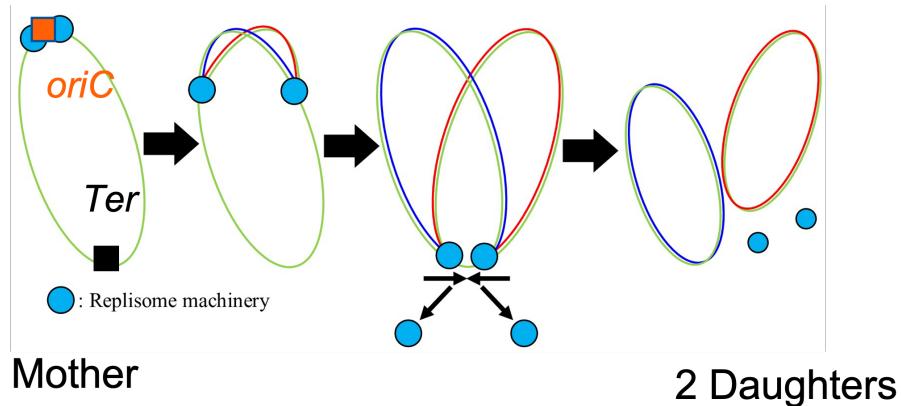


Figure 2.11: Ordered and Bidirectional Replication

The equivalent reaction to duplicate certain gene is in Table 2.3. The rate to replicate one gene in Table 2.4 is determined by a polymerization rate form proposed in [10].

Table 2.3: Reactions and Rates for Replication, Transcription, Translation and Degradation

Processes	Reactions	Kinetic Constant
Replication	$G_{\text{locusNum}} \rightarrow 2G_{\text{elongation}}$	$k_{\text{replication}}^{locusNum}$
Transcription	$RNA P + G_{\text{locusNum}} \rightarrow RNA P : G_{\text{locusNum}}$ $RNA P : G_{\text{locusNum}} \rightarrow R_{\text{locusNum}} + RNA P + G_{\text{locusNum}}$	$k_{\text{binding}}^{trsc}$ $s_{\text{trsc}}^{locusNum} k_{\text{trsc}}^{elongation}$
Translation	$Ribosome + R_{\text{locusNum}} \rightarrow Ribosome : R_{\text{locusNum}}$ $Ribosome : R_{\text{locusNum}} \rightarrow P_{\text{locusNum}} + Ribosome + R_{\text{locusNum}}$	$k_{\text{trans}}^{binding}$ $k_{\text{trans}}^{elongation}$
Degradation	$Degradosome + R_{\text{locusNum}} \rightarrow Degradosome : R_{\text{locusNum}}$ $Degradosome : R_{\text{locusNum}} \rightarrow NMPs + Degradosome$	$k_{\text{degra}}^{binding}$ $k_{\text{degra}}^{depoly}$

Transcription, translation and degradation are all depicted as a two-step binding and (de)polymerization reactions where the polymerization in transcription and translation shares the same rate form as that in replication. The rate for degradation from a mRNA to its monomers is calculated by dividing the monomer depletion rate over the length of mRNA.

Table 2.4: Rate Form for Replication, Transcription, Translation and Degradation

Processes	(de)polymerization Constants	Formulation
Replication	$k_{\text{replication}}^{\text{elongation}}$	$\frac{k_{\text{replication}}^{\text{cat}}}{[dNTP_1][dNTP_2] + \sum_i \frac{K_{D_i}}{[dNTP_i]} + L_{DNA} - 1}$
Transcription	$k_{\text{trsc}}^{\text{elongation}}$	$\frac{k_{\text{transcription}}^{\text{cat}}}{[NTP_1][NTP_2] + \sum_i \frac{K_{P_i}}{[NTP_i]} + L_{RNA} - 1}$
Translation	$k_{\text{trans}}^{\text{elongation}}$	$\frac{k_{\text{translation}}^{\text{cat}}}{[tRNA:aa_1][tRNA:aa_2] + \sum_i \frac{K_{D_i}}{[tRNA:aa_i]} + L_{protein} - 1}$
Degradation	$k_{\text{degra}}^{\text{depoly}}$	$\frac{k_{\text{degra}}^{\text{cat}}}{L_{mRNA}}$

Metabolism in ODE and Rate Law

As mentioned, replication, transcription, and translation all require monomers (deoxyribonucleotide (dNTPs), ribonucleotide (NTPs), and amino acids (AAs)) for the polymerization reactions.

All the supply of monomers comes from metabolism of JCVI-syn3A. The (d)NTPs are generated in nucleotide metabolism that transport extracellular nucleoside and convert nucleoside to (d)NTPs used in the DNA and RNA synthesis. The other building blocks, such as phosphoribosyl pyrophosphate (prpp), phosphoenolpyruvate (pep), and 1,3-Bisphosphoglyceric acid (1,3dpg) along these pathways are the products in glycolysis in central metabolism 2.12.

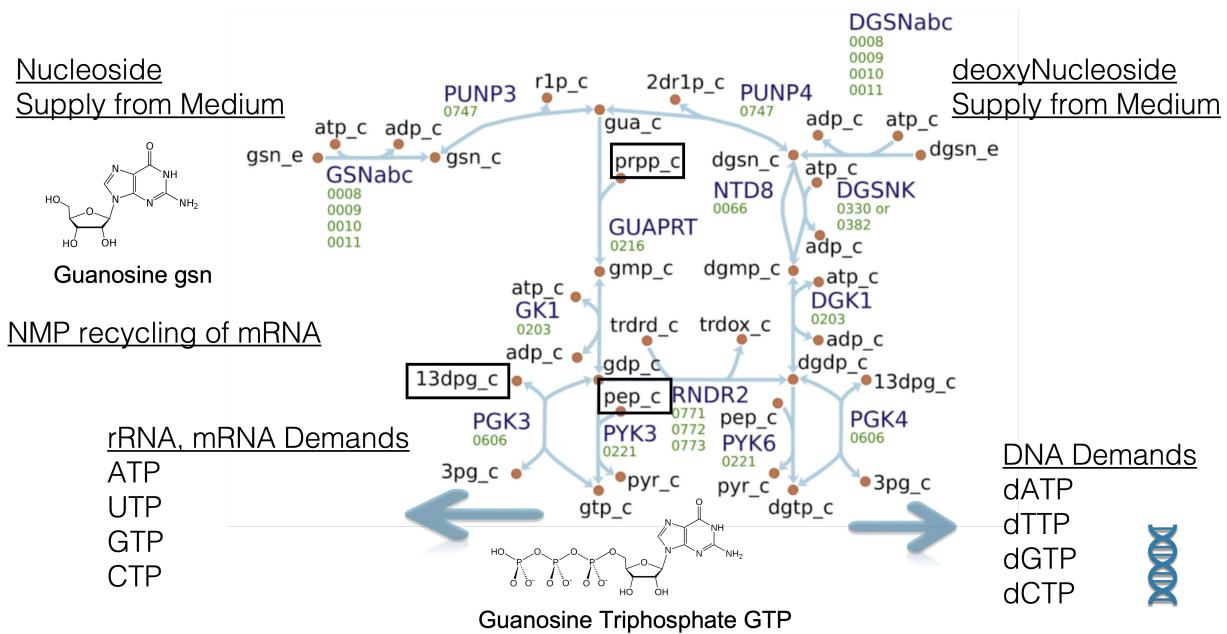


Figure 2.12: Supply of GTP and dGTP for DNA and RNA synthesis in nucleotide metabolism. The metabolites in the bracket come from central metabolism

The whole metabolism network can be separated to five connected sub-networks, including central, nucleotide, lipid, cofactor and amino acid. Glucose 6-phosphate (g6p) the immediate downstream of glucose in glycolysis connects central and lipid metabolism, while Adenosine triphosphate (ATP) that generated in glycolysis provides energy for reactions in all five sub-networks.

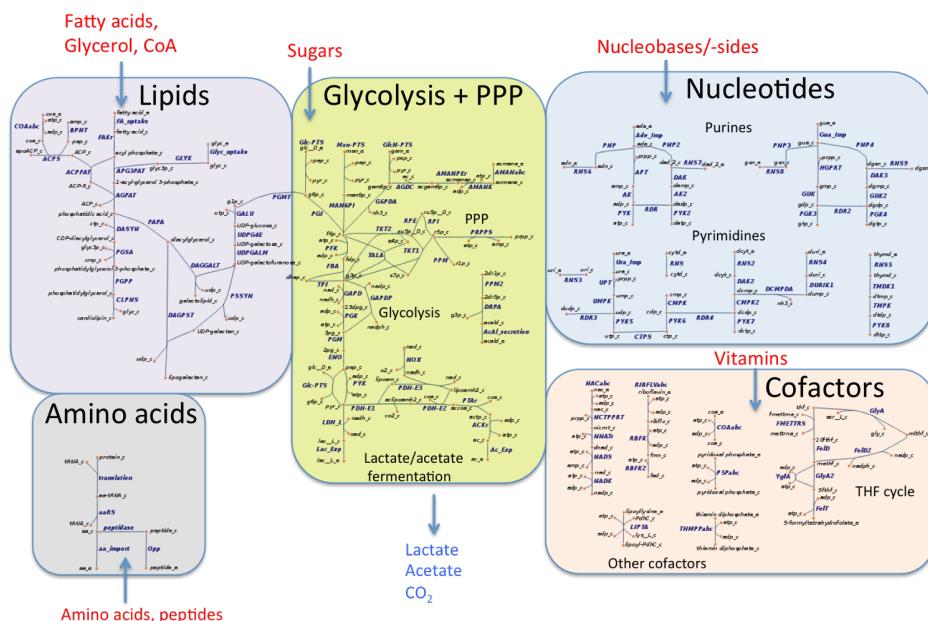


Figure 2.13: Whole Metabolism of JCVI-syn3A with five subsystems: central, nucleotide, lipid, cofactor and amino acid

In our current whole cell modeling, 197 metabolites are included, 148 cytoplasmic and 49 extra-cellular. 175 reactions connect the metabolites. Shown in the network of central metabolism, the orange nodes are the metabolite IDs. We use suffix '_c' to denote cytoplasmic and '_e' for extra-cellular. The blue arrows are reactions connecting different metabolites, with names in dark blue and related proteins' locusNums under the names.

ATP as the major energetic molecules energize cellular processes, without which the cell will die. In JCVI-syn3A, glycolysis is the only way to generate ATP. Phosphoenolpyruvate (pep) is an intermediate in both the upstream and downstream in glycolysis.

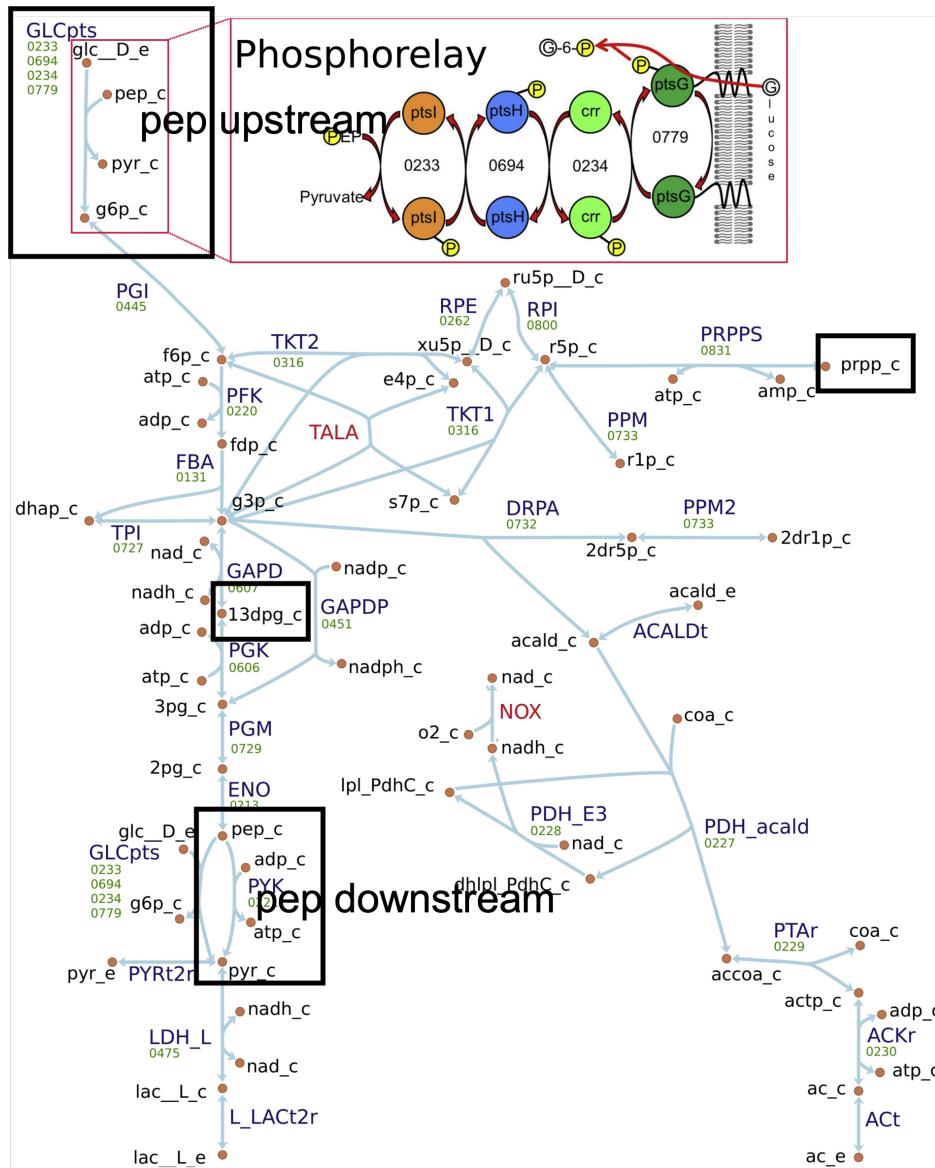
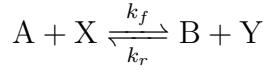


Figure 2.14: Central Metabolism in JCVI-syn3A. Phosphoenolpyruvate (pep) is both the upstream and downstream of glycolysis. 1,3-Bisphosphoglyceric acid (1,3dpg), Phosphoribosyl pyrophosphate (prpp) and pep are in nucleotide metabolism.

Most reactions in the metabolism requires certain proteins produced in genetic information pro-

cesses as enzymes or transporters. Convenience rate law and random binding model are used to depict the kinetics of these reactions [11].

The random binding model assumes the substrates bind to the enzyme/transporter in arbitrary order and are converted into the products, which then dissociate from the enzyme in arbitrary order. Convenience rate law further assumes the conversion step is the rate-determining step and the binding reactions are in quasi-equilibrium. For reaction



, the rate law is

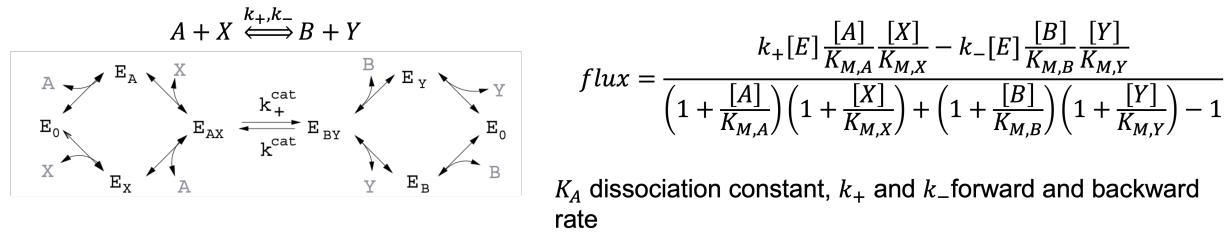


Figure 2.15: Mechanism of reaction $A + X \rightarrow B + Y$ in random binding model. $[E]$ is the concentration of enzyme, $[A]$ concentration of molecule A

Similar to genome information, the metabolic network is also stored in standard Systems Biology Markup Language (SBML) file. In the given SBML file of JCVI-syn3A, there are two compartments, cellular and extracellular with metabolites and reactions. However, SBML file itself does not store kinetic parameters and we choose to store them in a Excel file with multiple sheets for the simulation.

We can look at one reaction entry in the SBML file for more insights. Go to the SBML file and search DGSNabc. DGSNabc, the irreversible transport of deoxyguanosine (one of RNS ABC import system) in nucleotide metabolism for example, has three reactants and four products with four proteins involved. The geneProductAssociation is AND, meaning four proteins are all needed to perform this reactions. Biologically, four proteins are four sub-units of nucleoside ABC transporter.

2.3.3 Hybrid CME/ODE Algorithm

As mentioned in the Chemical Master Equation, the discreteness and stochasticity of chemical kinetics play a role when numbers of reactants are now significantly high. For the species in genetic information processes, gene's number approximately is 1 or 2 and for most proteins, their counts are less than 1000. Based on the initial volume of JCVI-syn3A (200 nm radius, 0.035 fL, 1 particle count equals 50 nM), the concentrations of genes are 50 or 100 nM, of proteins less than $50 \mu M$. All of these are dramatically lower than the concentrations of metabolites.

Therefore, it is necessary to simulate the kinetics in genetic information processes with stochastic chemical master equation (CME) and metabolism with deterministic ordinary differential equation (ODE).

Table 2.5: Concentrations of Species in Genetic Information Processes and Metabolism

	Numbers (Discrete)	Concentrations (Continuous)
G_0001	1 or 2	50 or 100 nM
R_0001	< 10	< 500 nM
P_0001	$100 \sim 10^3$	$5 \mu\text{M} \sim 50 \mu\text{M}$
Ribosome, degradosome	$100 \sim 10^3$	$5 \mu\text{M} \sim 50 \mu\text{M}$
ATP	$\sim 10^6$	$\sim 5 \text{ mM}$

To simulate the co-evolution of genetic information processes and metabolism, the communication needs to be performed to describe the interactions between these two subsystems.

Our current implementation of communication was first carried out in Yeast's galactose switch modeling published by our group in 2018 [12]. We first discretize the entire simulation length into piecewise communication time steps (hook intervals, t_H). During each communication time steps, t_H length CME simulation is performed to describe the kinetics in genetic information processes, followed by the communication from CME to ODE by passing the counts of proteins, consumption of monomers (dNTPs, NTPs, and AAs) and recycling of nucleoside monophosphates (NMPs) in mRNA degradation. Then t_H length ODE simulation is performed and the updated cell volume and concentrations of monomers passed to the CME.

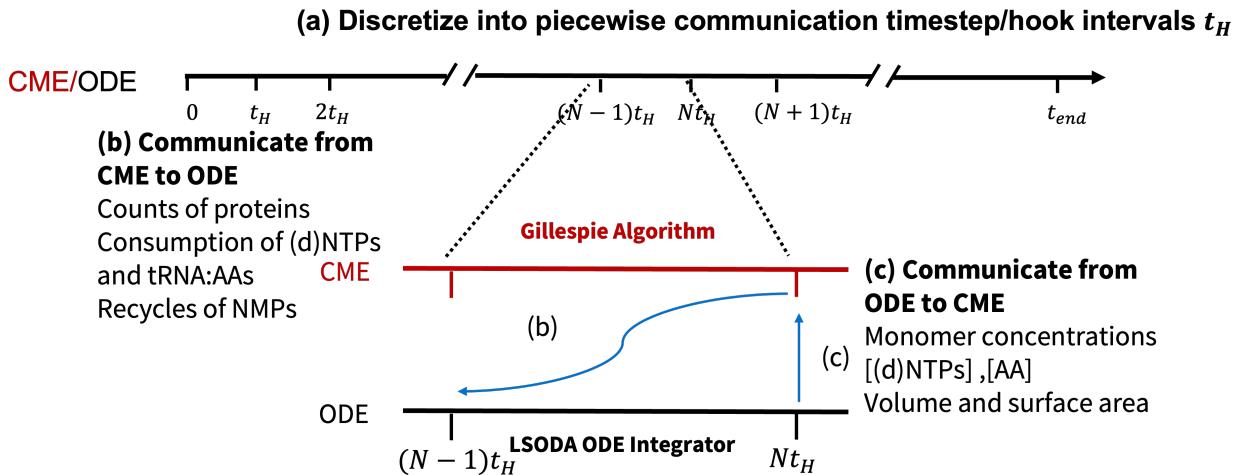


Figure 2.16: Communication Between CME and ODE to simulate the co-evolution of Genetic Information Processes and Metabolism.

In our current model, 6300 seconds' entire cell cycle is discretized with communication step 1 second.

2.3.4 Implementation in Coding

Four main input files are used in whole cell simulation, two Excel files containing initial counts/concentrations and reactions with kinetic parameters, one Genbank file and one SBML file. The

Genbank file contains the sequences and functions of genes, RNAs, and proteins. More details about how these files function in the model are available on [github](#).

Table 2.6: Input Files and Information Read In

File	File Format	Information Read IN
syn3A.gb	Genbank	Sequences and functions of genes, RNA and protein
Syn3A_updated.xml	SBML	Stoichiometric Coefficients
initial_concentrations.xlsx	Excel	Initial counts/concs of proteins, medium and metabolites
kinetic_params.xlsx	Excel	Reactions from SBML and kinetic parameters

We launch independent replicates to obtain the statistically significant cellular dynamics. The main script is **WCM_CMEODE_Hook.py** calling multiple other python scripts to construct and simulate the genetic information processes, metabolism and their interactions over the whole cell cycle.

Essentially, we use pyLM and build-in Gillespie algorithm to construct and simulate the genetic information processes. The ODEs are written/constructed by odecoll, a package developed by former group member and simulated by well-known LSODA algorithm in scipy.

The CME/ODE simulation is solely performed on CPUs, however NVIDIA GPUs and CUDA are needed to run Lattice Microbe. Using Intel(R) Xeon(R) Gold 6154 CPU @ 3.00GHz, each simulation will finish between 5 to 12 hours with 2 GB RAM usage. The variable time cost mainly comes from the adaptive time step LSODA ODE integrator.

The output files of current model are one log file and three CSV files respectively, recording counts of species in genetic information processes and metabolism, fluxes of metabolic reactions, and cell surface area and volume. General size for counts CSV file are 100 MB with approximate 5000 species and 6301 time points.

2.3.5 Results Analysis

Now run **pkl.ipynb** and **plots.ipynb** to do the analysis.

Emergent Cell Death

As mentioned, pep is an immediate in both upstream and downstream in glycolysis 2.14. If the concentration of pep goes to zero, the phosphorely (transport of glucose inside the cell) will halt, leading to the ATP concentration decrease that will finally affect all cellular processes. Since pep is both the upstream and downstream in glycolysis, the death is irreversible.

What is interesting is that the death is also stochastic, meaning the pep concentration goes to zero only happens in partial replicates. This strongly indicate that the death is related to the stochastic genetic information processes. And we are collaborating to using machine learning to disclose the highly non-linear dynamics of cell death.

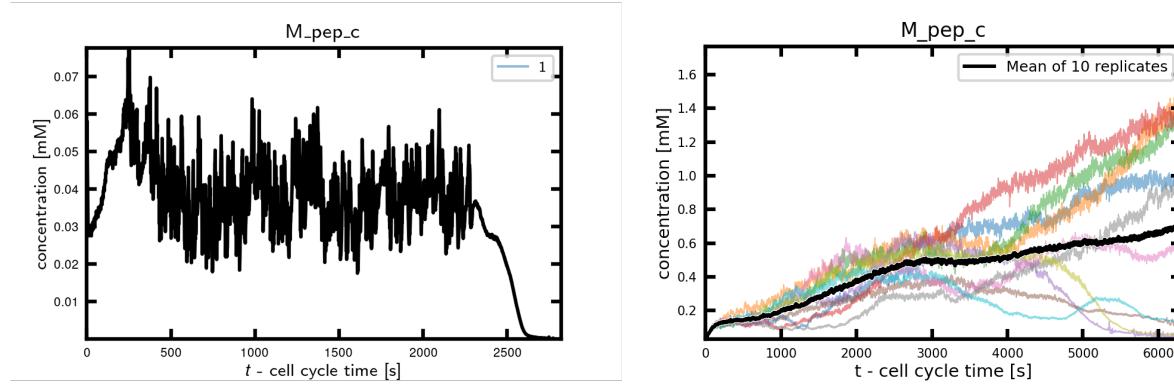


Figure 2.17: Traces of pep concentration. Left:Pep concentration goes to zero for one unhealthy/dead cell. Right:Pep concentration of 10 healthy replicates.

In the high degradation binding rate, the death happens in 48 replicates out of 100. While for low degradation binding rate, the death is 80 out of 100, much higher.

Cell Growth

Lipid molecules in metabolism and membrane proteins contributes to the membrane surface area. For the volume increase, we assume the cell remains spherical until the volume doubles. Based on the surface area, the radius then the volume is calculated. Most of the cells double the volume between 70 to 80 minutes.

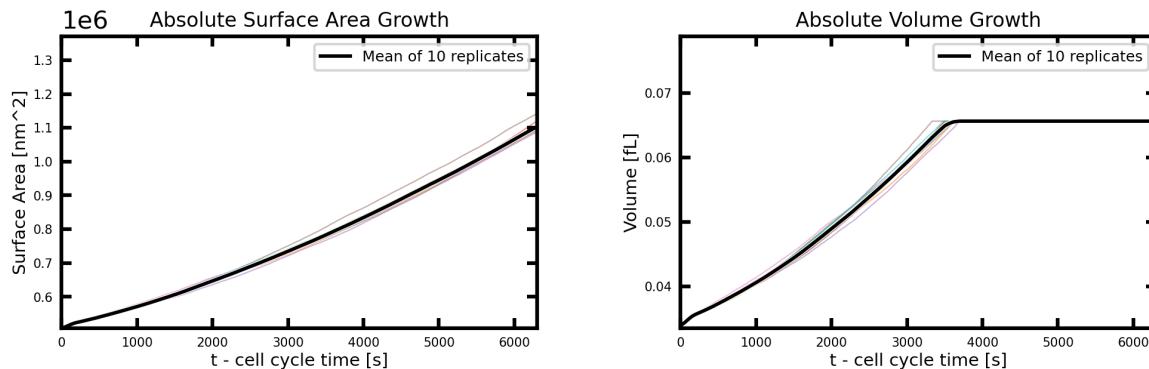


Figure 2.18: Cell surface area and volume increase.

Replication

For the replication, initiation can happen maximum 7 times during the whole cell cycle and the mean is 4.4. In this current model, we only allow replication initiation occurs after the replication finishes.

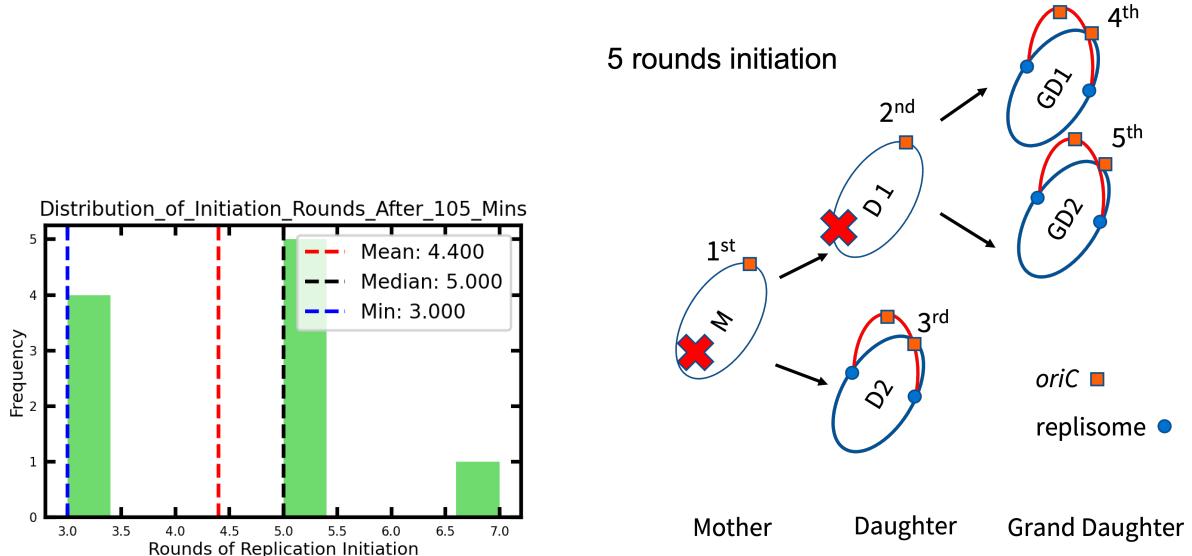


Figure 2.19: Distributions of rounds of initiation and the one possible theta structure of circular chromosomes of 5 rounds of replication initiation.

For 5 rounds of replication initiation, the data structure of chromosomes are shown in the right side of Figure 2.19. When the mother chromosome replication initiated and replicated, two daughter chromosomes was generated with no mother chromosome exists. Both daughter chromosomes initiates replication and one of two finishes the replication to generate two grand daughter chromosomes. And both two grand daughters initiates replications. So in this case, we will have three initialized but not finished replication in the cell, one daughter and two grand daughters chromosome.

The first round initiation finishes with 16 minutes with mean 4.4 minutes. And the filament length of DnaA on ssDNA can grow above 20 in Figure 2.20.

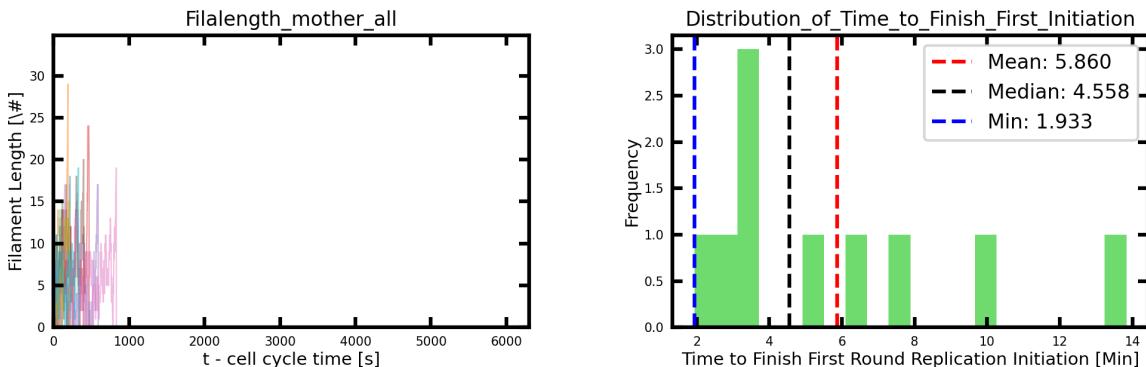


Figure 2.20: Distributions of protein counts. Left:Protein counts of 100 replicate at the end of the cell cycle. Right:Analytical Solution at steady state.

After the whole cell cycle, average gene copy number is 3.9 with minimal 2.8, meaning all genes are duplicated. It takes 40 to 60 minutes for the whole genome to be replicated.

As mentioned, the supply of nucleotide from metabolism is vital for the function of genetic information processes. In the DNA replication, the deoxythymidine triphosphate (dTTP) may encounter shortage, causing the pause of replication (See right upper picture in 2.24).

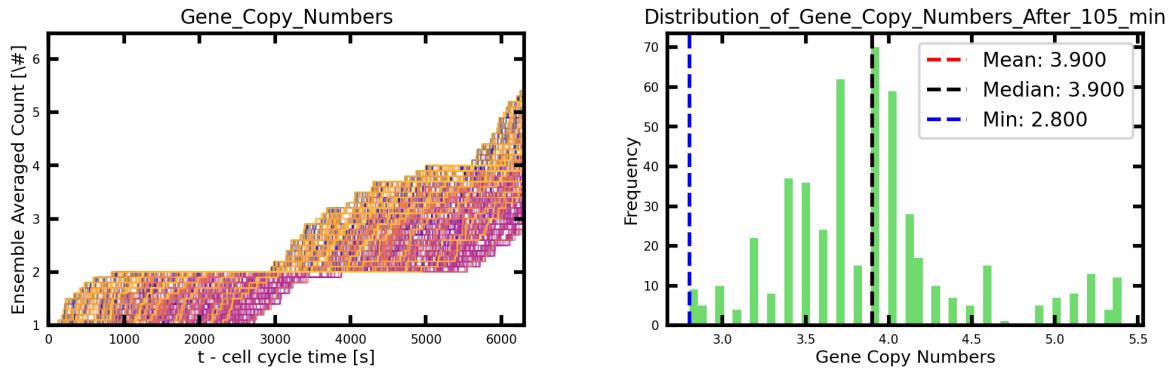


Figure 2.21: Gene Copy Numbers and the Distribution of Copy number at the end of the cell cycle.

Transcriptomics and Proteomics

In the transcriptomines, we will see that due to the binding of degradosome and ribosomes, mRNAs seldom exist as free mRNA but under translation or degradation. Along with the increase of gene numbers, the count of all mRNAs (free, bound to degradosome, and to ribosome) also increase and doesn't reach steady state.

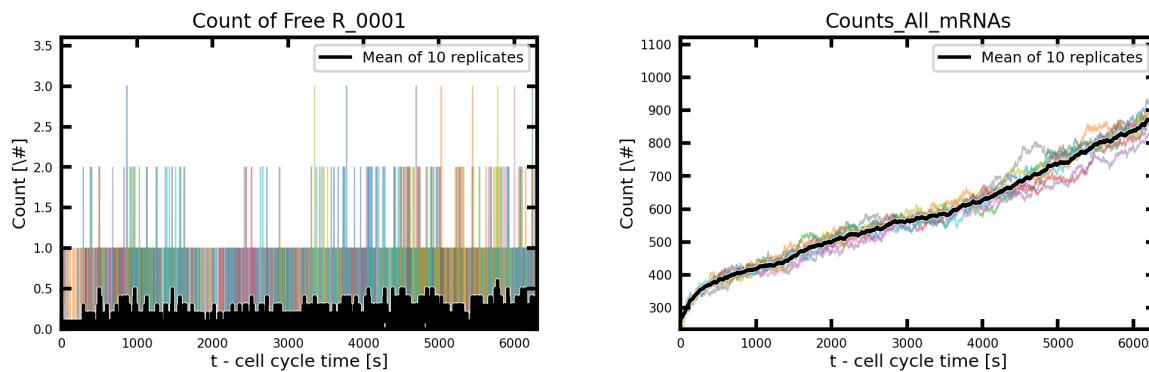


Figure 2.22: Traces of free mRNA, R_0001 of 10 replicates and the number of all mRNAs over the cell cycle.

Due to the fact that no degradation of protein considered in our current model, the protein count accumulates over time. Most proteins roughly double after one cell cycle.

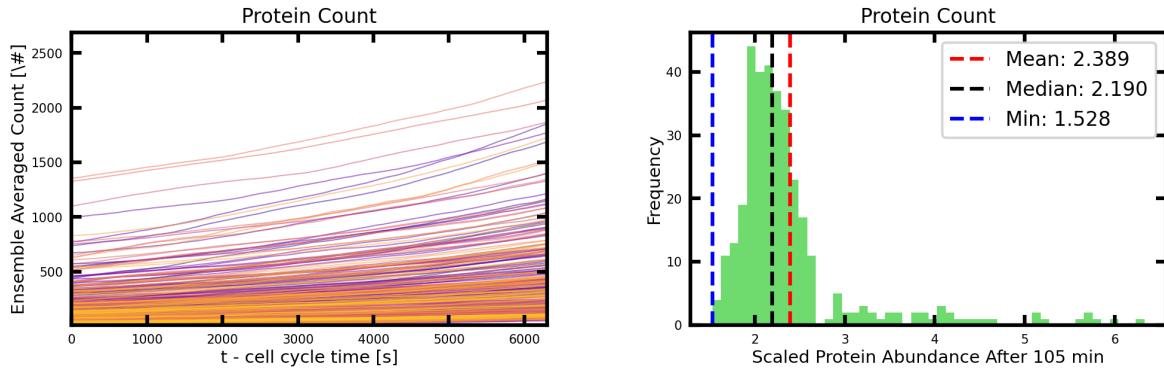


Figure 2.23: Ensemble averaged count of different proteins over the cell cycle and the distribution of scaled protein abundances.

Traces of ALL Metabolites

From our current simulation, we can also easily plot all traces of metabolites over the whole cell cycle.

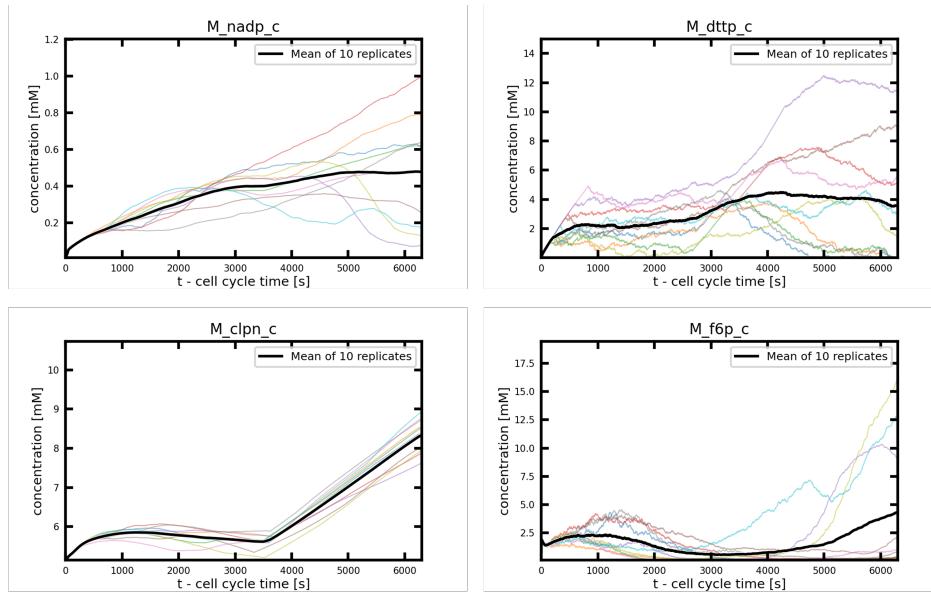


Figure 2.24: Traces of Four Metabolites Over the Entire Cell Cycle

Chapter 3

License and Copyright

University of Illinois Open Source License
Copyright © 2008-2024 Luthey-Schulten Group, All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the Software), to deal with the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimers.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimers in the documentation and/or other materials provided with the distribution.
- Neither the names of the Luthey-Schulten Group, University of Illinois at Urbana-Champaign, nor the names of its contributors may be used to endorse or promote products derived from this Software without specific prior written permission.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS WITH THE SOFTWARE.

Bibliography

- [1] Fujikawa, N, Kurumizaka, H, Nureki, O, Terada, T, Shirouzu, M, Katayama, T, Yokoyama, S (2003) Structural basis of replication origin recognition by the DnaA protein. *Nucleic Acids Research* 31:2077–2086.
- [2] Duderstadt, KE, Chuang, K, Berger, JM (2011) DNA stretching by bacterial initiators promotes replication origin opening. *Nature* 478:209–213 Publisher: Nature Publishing Group.
- [3] Peterson, J, Hallock, M, Cole, J, Luthey-Schulten, Z (2013) A Problem Solving Environment for Stochastic Biological Simulations. *Proceedings High Performance Computing Networking, Storage and Analysis Companion (SCC)*.
- [4] Thornburg, ZR, Bianchi, DM, Brier, TA, Gilbert, BR, Earnest, TM, Melo, MC, Safronova, N, Sáenz, JP, Cook, AT, Wise, KS, Hutchison, CA, Smith, HO, Glass, JI, Luthey-Schulten, Z (year?) Fundamental behaviors emerge from simulations of a living minimal cell. 185:345–360.e28.
- [5] Thornburg, ZR, Melo, MCR, Bianchi, D, Brier, TA, Crotty, C, Breuer, M, Smith, HO, Hutchison, CA, Glass, JI, Luthey-Schulten, Z (2019) Kinetic Modeling of the Genetic Information Processes in a Minimal Cell. *Frontiers in Molecular Biosciences* 6.
- [6] Shahrezaei, V, Swain, PS (year?) Analytical distributions for stochastic gene expression. 105:17256–17261.
- [7] Hutchison, CA, Chuang, RY, Noskov, VN, Assad-Garcia, N, Deerinck, TJ, Ellisman, MH, Gill, J, Kannan, K, Karas, BJ, Ma, L, Pelletier, JF, Qi, ZQ, Richter, RA, Strychalski, EA, Sun, L, Suzuki, Y, Tsvetanova, B, Wise, KS, Smith, HO, Glass, JI, Merryman, C, Gibson, DG, Venter, JC (2016) Design and synthesis of a minimal bacterial genome. *Science* 351:aad6253 Publisher: American Association for the Advancement of Science.
- [8] Breuer, M, Earnest, TM, Merryman, C, Wise, KS, Sun, L, Lynott, MR, Hutchison, CA, Smith, HO, Lapek, JD, Gonzalez, DJ, de Crécy-Lagard, V, Haas, D, Hanson, AD, Labhsetwar, P, Glass, JI, Luthey-Schulten, Z (2019) Essential metabolism for a minimal cell. *eLife* 8:e36842 Publisher: eLife Sciences Publications, Ltd.
- [9] Earnest, TM, Cole, JA, Peterson, JR, Hallock, MJ, Kuhlman, TE, Luthey-Schulten, Z (2016) Ribosome biogenesis in replicating cells: Integration of experiment and theory. *Biopolymers* 105:735–751 _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/bip.22892>.

- [10] Hofmeyr, JHS, Gqwaka, OP, Rohwer, JM (2013) A generic rate equation for catalysed, template-directed polymerisation. *FEBS Letters* 587:2868–2875 eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1016/j.febslet.2013.07.011>.
- [11] Liebermeister, W, Klipp, E (2006) Bringing metabolic networks to life: convenience rate law and thermodynamic constraints. *Theoretical Biology and Medical Modelling* 3:41.
- [12] Bianchi, DM, Peterson, JR, Earnest, TM, Hallock, MJ, Luthey-Schulten, Z (year?) Hybrid CME–ODE method for efficient simulation of the galactose switch in yeast. 12:170–176.