Short Communication
Open Access

AdvanceSyn Toolkit: An open-source suite for model development and analysis in biological engineering

Abstract

Modelling and simulations are useful means to screen potential experimental designs for metabolic engineering. Genome-scale models of metabolism (GSM) and kinetic models (KMs) are the two main approaches for modelling, which resulted in largely disjoint computational tools for GSMs and KMs. Existing tools for GSMs require knowledge of the underlying programming languages while the development and merger of two or more KMs is difficult. In this work, AdvanceSyn Toolkit is an open-sourced high-level command-line tool to develop KMs, and to analyse GSMs and KMs; licensed under the Apache License, Version 2.0, for academic and not-for-profit use. It elevates the need to know the underlying programming language for GSM analysis. AdvanceSyn Model (ASM) specification is a simple and modular format for model development and AdvanceSyn Toolkit provides a method to merge two or more model files for simulation and sensitivity analysis.

Volume 9 Issue 4 - 2020

Maurice HT Ling
AdvanceSyn Private Limited, Republic of Singapore, Singapore

Correspondence: Maurice HT Ling, AdvanceSyn Private Limited, Republic of Singapore, Singapore, Email mauriceling@advancesyn.com

Received: September 22, 2020 | **Published:** October 08, 2020

Introduction

Metabolic engineering and synthetic biology are important complementary platforms in the current Fourth Industrial Revolution to translate research into commercially viable products,[1,2] with many researchers calling for the utilization of both fields[3–5] to access nature's diversity.[6] As engineered circuits are becoming increasingly complex, our limited knowledge in optimizing complex circuits often impedes current efforts and computational models are seen as a means of screening potential designs.[7] Several recent studies had used computational models to aid in designing engineering strategies. For example, Kim et al.[8] used genome-scale metabolic model (GSM) to improve lipid production in *Yarrowia lipolytica* and Wayman et al.[9] used GSM to improve glycan production in *Escherichia coli* while Tian et al.[10] used kinetic model (KM) to improve N-acetylneuraminic acid production in *Bacillus subtilis*.

Modelling refers to the mathematical construction of a model while simulation is the execution and solving of the model.[11] GSMs and KMs are the two main modelling approaches for metabolic engineering. GSMs are concerned with the distribution of intracellular fluxes of metabolites while KMs are concerned with the interactions of metabolites. GSMs, also known as constraint-based models, are based largely on metabolic stoichiometries and mass balance[12] while KMs are largely based on rate equation.[13] As such, GSMs cannot capture the relationship between flux, enzyme expression, metabolite levels, and regulation that is possible with KMs.[14] Hence, GSMs generally provides only steady-state distribution of metabolic fluxes while KMs provide time series of metabolite concentrations. Kerkhoven et al.[15] surmised GSMs as top-down approach while KMs as bottom-up approach. Sier et al.[16] has demonstrated that coupling both approaches can yield novel insights.

Due to the fundamental differences in modelling philosophies, computational tools for GSMs and KMs are largely disjoint. The core tool for GSM is COBRA Toolbox[17] for MATLAB and its subsequent version, COBRApy[18] for Python programming language. A large repertoire of tools had been developed for GSMs over the last 2 decades[19] since the publication of the first GSM in 1999.[20] Cameo[21] builds on top of these tools and presents a high-level interface for GSM usage. Yet, Python programming knowledge is required to use Cameo as it is a Python library. On the KM front, the most well-known tool is COPASI[22] which had been used in many studies.[23] However, despite providing a user-friendly interface to simulate models and present results, it is difficult to merge multiple existing models in COPASI as it requires finding the common metabolites between the two models and rewriting the affected equations.

To address these difficulties, AdvanceSyn Toolkit is presented as a high-level command-line tool to develop KMs, and to analyse GSMs and KMs. AdvanceSyn Toolkit wraps key operations in Cameo[21] into a unified command-line interface; thus, elevating the need to know Python programming. As a command-line interface tool, AdvanceSyn Toolkit can be incorporated into computational biology and bioinformatics pipelines.[24] AdvanceSyn Model (ASM) specification is based on Antimony language[25] used in Tellurium,[26] which is simple and modular; and initialization file structure. This makes ASM a code file format rather than a data-exchange file format; such as JSON, which is substantially more verbose, requires more structure, and data type formatting. Moreover, AdvanceSyn Toolkit provides a method to merge two or more ASM files, a feature not found in existing tools, which allows for more effective reuse of existing models.

Results

AdvanceSyn Toolkit is open source software written in Python and Python-Fire module (https://github.com/google/python-fire), which aims to simplify the implementation of command-line interface in Python 3. This combination has been used in several other tools.[27,28] AdvanceSyn Toolkit has two main sets of operations (Figure 1; Supplementary materials S2 to S21); namely, 10 operations for KMs, and 9 operations for GSMs via Cameo.[21] Here, three use cases are presented to illustrate core features of AdvanceSyn Toolkit.

Use Case #1; Development and Analysis of KM: Here, two separate KMs were developed, one for glycolysis pathway (Supplementary material S22) and another for pentose phosphate pathway (Supplementary material S23). Glycolysis pathway consists of nine reaction steps while pentose phosphate pathway consists of six reaction steps (Figure 2). However, there is one reaction linking glycolysis, from glucose-6-phosphate (g6p), to 6-phosphogluconolactone (pgl6) in pentose phosphate pathway. For simplicity, co-factor(s) and co-substrate(s) such as ATP and NADP are not modelled. Each reaction step is mathematically modelled as a rate expression, also known as

rate law,[29] which corresponds to the kinetic law in System Biology Markup Language (SBML).[30] This allows the concentration of each metabolite is modelled as a rate equation in the form of

$$\frac{d[metabolite]}{dt} = \sum_{i=1}^{N} production_i - \sum_{i=1}^{N} usage_i$$

where each production and usage term represents a reaction step and in this use case, is modelled as a product of the concentration of enzyme and substrate(s) in the general form of $[enzyme] \times \sum_{i=1}^{N} [substrate_i]$.

For example, the concentration of glucose-6-phosphate (g6p) over time is modelled as $\frac{d[g6p]}{dt} = [HK][glucose] - ([PGI][g6p] + [G6PD][g6p])$ where HK, PGI, and G6PD are hexokinase (converting glucose to glucose-6-phosphate), phosphoglucoisomerase (converting glucose-6-phosphate to fructose-6-phosphate), and glucose 6-phosphate dehydrogenase (converting glucose-6-phosphate to 6-phosphogluconolactone), respectively. AdvanceSyn Toolkit does not assume any specific type of rate expression as each reaction is defined by the user in the model, making it possible to have a different type of rate expression for each reaction.

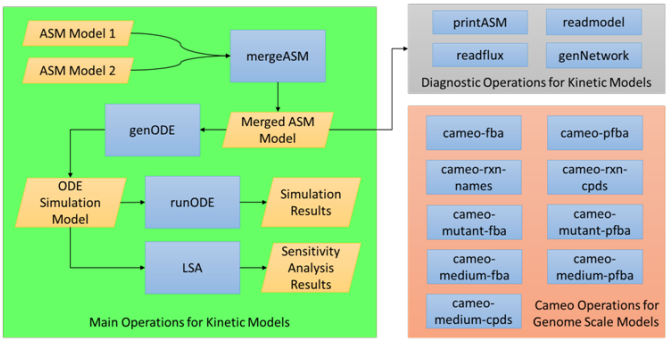

Figure 1 Overview of functionality.

The concentration of all metabolites and enzymes are fixed at 1 micromolar; except glucose, which is given at 1 millimolar. Glycolysis pathway model and pentose phosphate pathway model were merged (Supplementary material S24) based on common metabolites across the two models before generating a simulation script (Supplementary material S25) for simulation (Supplementary material S26). Merging is performed under the assumption that units used in both models are identical. Briefly, the models are merged using the union set of the metabolites as nodes and re-coding each reaction as edges and parameters as attributes. Our simulation results suggest that the rate of 6-phosphogluconolactone (pgl6) increases faster than both pyruvate (pyr) and xylulose-5-phosphate (x5p) given the current conditions. Sensitivity analysis (Supplementary material S27) using One-Factor-at-a-Time method[31,32] was used to evaluate the effects of varying enzyme concentrations (as listed in the Variables section of the model) on the overall metabolite distribution. Mean square error (MSE) between the metabolites in each variation of enzyme concentration and that of the original model, which is directly proportional to the impact of the enzyme concentration on the overall metabolite distribution, were calculated as

$$MSE_{Enzyme} = \sum_{i=1}^{N} \left(metabolite_{[Enzyme_{changed}],i} - metabolite_{[Enzyme_{original}],i} \right)^2 / N \ .$$

Our sensitivity analysis suggests that hexokinase concentration has the most impact followed by phosphoglucoisomerase and glucose 6-phosphate dehydrogenase. The results of both simulation and sensitivity analysis are given as comma-delimited files to enable analysis by other statistical software. This also allows the user to determine the effect of the concentration of each enzyme on a metabolite of interest, which can be useful in informing engineering strategies (Figure 2).[33]

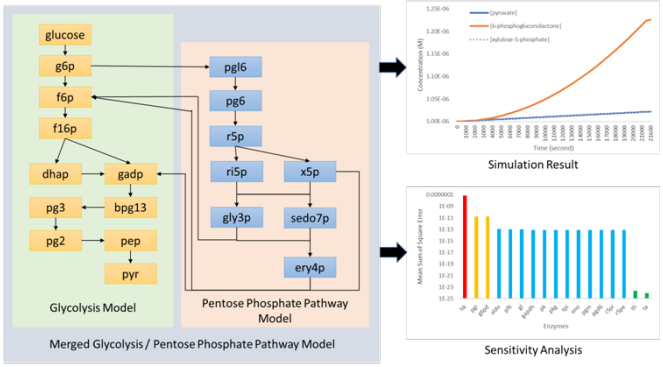

Figure 2 Merging and analysis of KMs. The acronyms for metabolites in the models are as follow (in alphabetical order): bpg13 (D-1,3-bisphosphoglycerate), dhap (Dihydroxyacetone-phosphate), ery4p (erythose-4-phosphate), g6p (D-Glucose-6-phosphate), gadp (D-glyceraldehyde-3-phosphate), glucose (D-glucose), gly3p (glyceraldehyde-4-phosphate), f16p (D-Fructose-1,6-bisphosphate), f6p (D-Fructose-6-phosphate), pep (phosphoenolpyruvate), pg2 (2-phosphoglycerate), pg3 (3-phosphoglycerate), pg6 (6-phosphogluconate), pgl6 (6-phosphogluconolactone), pyr (pyruvate), r5p (ribulose-5-phosphate), ri5p (ribose-5-phosphate), sedo7p (sedoheptulose-7-phosphate), and x5p (xylulose-5-phosphate) (Figure 3).

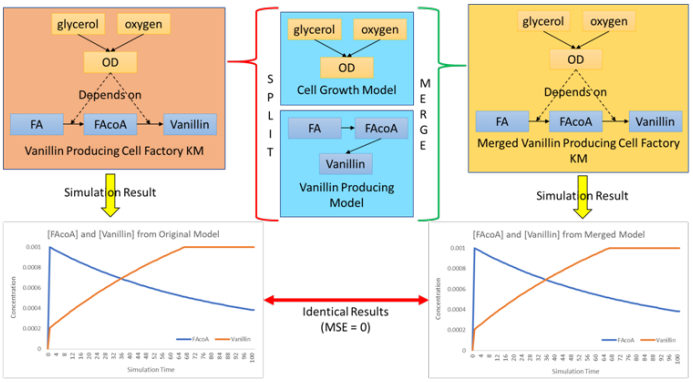

Figure 3 Splitting and merging of model gives identical simulation results.

Use Case #2; Merging of KMs: To further emphasize merging of KMs, the simplified vanillin producing cell factory KM by Yeoh et al.[34] was implemented in totality (Supplementary material S28) and simulated. The original model was split into two constituent models of cell growth model (Supplementary material S29) and vanillin producing model (Supplementary material S30). These two constituent models were then merged (Supplementary material S31) and simulated. Simulation results from the original and merged models were compared. Our result (Figure 3) shows that simulation results from the original model is identical (MSE = 0) to that of the merged model, demonstrating that the model merging algorithm is functioning correctly. In addition, this also demonstrate an important usage of AdvanceSyn Toolkit where cloned gene cassette(s), vanillin producing pathway in this case, can be merged with cell host model. Moreover, Yeoh et al.[34] use Hill equation,[35] which is different to that in Use Case #1; this further emphasizes that AdvanceSyn Toolkit does not assume any specific type of rate expression.

Use Case #3; Analysis of GSM: GSMs are commonly used for evaluating knockout strategies[36] or media components[37] to optimize yield of a metabolite of interest. AdvanceSyn Toolkit is built on top of

the functions of Cameo[21] to evaluate the biomass objective function,[38] which is a proxy for growth rate, or fluxes when one or more media component(s) or enzyme expression(s) were changed. Chang and Ling[39] had used AdvanceSyn Toolkit to explain the effects of varying glucose concentration in media, using cameo-medium-fba or cameo-medium-pfba operations, to the fluxes of *Escherichia coli* MG1655 using the GSM model, iAF1260.[40] It is essentially an attempt to explain Monod Equation in terms of metabolism and found a strong correlation (r = 0.972) between Monod's predicted growth rate and biomass objective value from GSM. Moreover, Chang and Ling[39] suggests that Monod's predicted growth rate of *E. coli* MG1655 can be predicted by the fluxes of 14 enzymes. This illustrates that AdvanceSyn Toolkit can use operations from Cameo[21] for GSM analysis without the need to learn Python programming.

Conclusion

AdvanceSyn Toolkit is an open-source, high-level command-line tool to develop KMs, and to analyse GSMs and KMs. The ability to merge two or more KMs into a unified KM is a unique feature of AdvanceSyn Toolkit as this feature allows for incremental development of more advanced models. AdvanceSyn Toolkit is a project under active development. However, it is not possible to merge GSMs with KMs or to merge multiple GSMs at this moment. Other future work includes interfacing with existing tools to accept a diverse range of model specifications in order to be an effective model translator, and to expand the repertoire of existing tools by chaining operations across various existing tools.

Supplementary materials and data

The supplementary materials and data set for this article are available at http://bit.ly/ADSToolkit_Supplement and http://bit.ly/ADSToolkit_DataSet respectively.

Availability and license

AdvanceSyn Toolkit is licensed under the Apache License, Version 2.0 for academic and not-for-profit use only, and is available at https://bit.ly/ADSToolkit.

Acknowledgments

None.

Conflicts of interest

The author is a co-founder of AdvanceSyn Private Limited and AdvanceSyn Toolkit is employed in consultancy services offered by the company.

References

1. Ramzi AB. Metabolic Engineering and Synthetic Biology. *Adv Exp Med Biol*. 2018;1102:81–95.
2. Palazzotto E, Tong Y, Lee SY, Weber T. Synthetic Biology and Metabolic Engineering of Actinomycetes for Natural Product Discovery. *Biotechnol Adv*. 2019;37(6):107366.
3. Choi KR, Jang WD, Yang D, et al. Systems Metabolic Engineering Strategies: Integrating Systems and Synthetic Biology with Metabolic Engineering. *Trends Biotechnol.* 2019;37(8):817–837.
4. Nguyen GT, Kim Y-G, Ahn J-W, et al. Structural Basis for Broad Substrate Selectivity of Alcohol Dehydrogenase YjgB from Escherichia coli. *Molecules*. 2020:25(10):E2404.
5. Shen Y-P, Niu F-X, Yan Z-B, et al. Recent Advances in Metabolically Engineered Microorganisms for the Production of Aromatic Chemicals Derived From Aromatic Amino Acids. *Front Bioeng Biotechnol*. 2020;8:407.
6. King JR, Edgar S, Qiao K, et al.. Accessing Nature's Diversity through Metabolic Engineering and Synthetic Biology. *F1000Research*. 2016;5:F1000 Faculty Rev-397.
7. Naseri G, Koffas MAG. Application of Combinatorial Optimization Strategies in Synthetic Biology. *Nat Commun*. 2020;11(1):2446.
8. Kim M, Park BG, Kim E-J, et al. In Silico Identification of Metabolic Engineering Strategies for Improved Lipid Production in Yarrowia lipolytica by Genome-Scale Metabolic Modeling. *Biotechnol Biofuels*. 2019;12:187.
9. Wayman JA, Glasscock C, Mansell TJ, et al. Improving Designer Glycan Production in Escherichia coli through Model-Guided Metabolic Engineering. *Metab Eng Commun*. 2019;9:e00088.
10. Tian R, Liu Y, Chen J, et al. Synthetic N-terminal Coding Sequences for Fine-Tuning Gene Expression and Metabolic Engineering in Bacillus subtilis. *Metab Eng*. 2019;55:131–141.
11. Ling M. Of (Biological) Models and Simulations. *MOJ Proteomics Bioinforma*. 2016;3:00093.
12. Gu C, Kim GB, Kim WJ, Kim HU, Lee SY. Current Status and Applications of Genome-Scale Metabolic Models. *Genome Biol*. 2019;20(1):121.
13. Resat H, Petzold L, Pettigrew MF. Kinetic Modeling of Biological Systems. *Methods Mol Biol*. 2009;541:311–335.
14. Strutz J, Martin J, Greene J, et al. Metabolic Kinetic Modeling Provides Insight into Complex Biological Questions, but Hurdles Remain. *Curr Opin Biotechnol.* 2019;59:24–30.
15. Kerkhoven EJ, Lahtvee P-J, Nielsen J. Applications of Computational Modeling in Metabolic Engineering of Yeast. *FEMS Yeast Res*. 2015;15(1):1–13.
16. Sier JH, Thumser AE, Plant NJ. Linking Physiologically-Based Pharmacokinetic and Genome-Scale Metabolic Networks to Understand Estradiol Biology. *BMC Syst Biol*. 2017;11(1):141.
17. Schellenberger J, Que R, Fleming RMT, et al. Quantitative Prediction of Cellular Metabolism with Constraint-Based Models: The COBRA Toolbox v2.0. *Nat Protoc*. 2011;6(9):1290–307.
18. Ebrahim A, Lerman JA, Palsson BO, et al. COBRApy: COnstraints-Based Reconstruction and Analysis for Python. *BMC Syst Biol*. 2013;8;7:74.
19. Machado D, Herrgård MJ. Co-Evolution of Strain Design Methods Based on Flux Balance and Elementary Mode Analysis. *Metab Eng Commun*. 2015;2:85–92.
20. Edwards JS, Palsson BO. Systems Properties of the Haemophilus influenzae Rd Metabolic Genotype. *J Biol Chem*. 1999;274(25):17410–17416.
21. Cardoso JGR, Jensen K, Lieven C, et al. Cameo: A Python Library for Computer Aided Metabolic Engineering and Optimization of Cell Factories. *ACS Synth Biol*. 2018;7(4):1163–1166.
22. Hoops S, Sahle S, Gauges R, et al. COPASI — A COmplex PAthway SImulator. *Bioinformatics*. 2006;22(24):3067–3074.
23. Bergmann FT, Hoops S, Klahn B, et al. COPASI and Its Applications in Biotechnology. *J Biotechnol*. 2017;261:215–220.
24. Leipzig J. A Review of Bioinformatic Pipeline Frameworks. *Brief Bioinform*. 2017;18(3):530–536.

25. Smith LP, Bergmann FT, Chandran D. Antimony: A Modular Model Definition Language. *Bioinformatics*. 2009;25(18):2452–2454.

26. Choi K, Medley JK, König M, et al. Tellurium: An Extensible Python-Based Modeling Environment for Systems and Synthetic Biology. *Biosystems*. 2018;171:74–79.

27. Ling MH. Island: A Simple Forward Simulation Tool for Population Genetics. *Acta Sci Comput Sci*. 2019;1(2):20–22.

28. Ling MH. RANDOMSEQ: Python Command–Line Random Sequence Generator. *MOJ Proteomics Bioinforma*. 2018;7(4):206–208.

29. Liebermeister W, Uhlendorf J, Klipp E. Modular Rate Laws for Enzymatic Reactions: Thermodynamics, Elasticities and Implementation. *Bioinformatics*. 2010;26(12):1528–1534.

30. Hucka M, Bergmann FT, Hoops S, et al. The Systems Biology Markup Language (SBML): Language Specification for Level 3 Version 1 Core. *J Integr Bioinforma*. 2015;12(2):266.

31. Abou-Taleb KA, Galal GF. A Comparative Study Between One-Factor-At-A-Time and Minimum Runs Resolution-IV Methods for Enhancing the Production of Polysaccharide by Stenotrophomonas daejeonensis and Pseudomonas geniculate. *Ann Agric Sci*. 2018;63(2):173–180.

32. Razavi S, Gupta HV. What Do We Mean by Sensitivity Analysis? The Need for Comprehensive Characterization of "Global" Sensitivity in Earth and Environmental Systems Models: A Critical Look at Sensitivity Analysis. *Water Resour Res*. 2015;51(5):3070–3092.

33. Tamano K. Enhancing Microbial Metabolite and Enzyme Production: Current Strategies and Challenges. *Front Microbiol*. 2014;5:718.

34. Yeoh JW, Jayaraman S, Tan SG-D, et al. A Model-Driven Approach towards Rational Microbial Bioprocess Optimization. *Biotechnol Bioeng*. 2020;10.1002/bit.27571.

35. Hill AV. The Combinations of Haemoglobin with Oxygen and with Carbon Monoxide. *I Biochem J*. 1913;7(5):471–480.

36. Nair G, Jungreuthmayer C, Zanghellini J. Optimal Knockout Strategies in Genome-Scale Metabolic Networks Using Particle Swarm Optimization. *BMC Bioinformatics*. 2017;18(1):78.

37. Li C-T, Yelsky J, Chen Y, et al. Utilizing Genome-Scale Models to Optimize Nutrient Supply for Sustained Algal Growth and Lipid Productivity. *Npj Syst Biol Appl*. 2019;5(1):33.

38. Feist AM, Palsson BO. The Biomass Objective Function. *Curr Opin Microbiol*. 2010;13(3):344–349.

39. Chang ED, Ling MH. Explaining Monod in terms of Escherichia coli metabolism. *Acta Sci Microbiol*. 2019;2(9):66–71.

40. Feist AM, Henry CS, Reed JL, et al. A Genome-Scale Metabolic Reconstruction for Escherichia coli K-12 MG1655 that Accounts for 1260 ORFs and Thermodynamic Information. *Mol Syst Biol*. 2007;3:121.

Citation: Ling MHT. AdvanceSyn Toolkit: An open-source suite for model development and analysis in biological engineering. *MOJ Proteomics Bioinform.* 2020;9(4):83–86. DOI: 10.15406/mojpb.2020.09.00286