LOICA: Logical Operators for Integrated Cell Algorithms

Gonzalo Vidal[‡], Carlos Vidal Céspedes[‡], Guillermo Yáñez Feliú[†], and Timothy J. Rudge[*,‡,†].

[‡]Institute for Biological and Medical Engineering, Schools of Engineering, Biology and Medicine, Pontificia Universidad Católica de Chile, Santiago, Chile
[†]Department of Chemical and Bioprocess Engineering, School of Engineering, Pontificia Universidad Católica de Chile, Santiago, Chile

[*]Corresponding author. Email: trudge@uc.cl

Background

Synthetic Biology is an interdisciplinary field that mixes life sciences and engineering. From this perspective life is an object to engineer, and a rational way to engineer cells is through its blueprint or DNA. This can be done by introducing synthetic DNA that encodes a synthetic regulatory network, also known as "genetic circuit". The design-build-test-learn (DBTL) cycle is central to engineering disciplines and each phase requires appropriate tools, standards and workflows, which are still in development in synthetic biology. The synthetic biology open language (SBOL) is an open standard for the representation of *in silico* biological designs that covers the DBTL cycle and has attracted a community of developers that have produced an ecosystem of software tools (McLaughlin et al., 2020). Software tools based on a common standard are an effective way to transfer knowledge, making available and accessible research results. They also create different abstraction levels that allow for example researchers working on genetic systems at device level to use the knowledge generated by others working at part level. In this sense synthetic biology aims for the same reliability in incremental and modular design that takes us from a transistor to a computer or a rocket.

Modelling is key in the DBTL cycle and is mainly related to the design and learn stages; a model states a well-defined hypothesis about the system and is the basis for computer aided design (CAD) which accelerates and automatizes the design process. Genetic circuits are dynamical systems, however current state-of-the-art genetic circuit design tools model them in steady-state as logic circuits (Nielsen et al., 2016). In this view circuits transition between discrete high and low steady-state gene expression levels according to input signal concentrations (Shin, Zhang, Der, Nielsen, & Voigt, 2020). In contrast, dynamical systems (such as the repressilator) are autonomous and follow continuous non-steady-state dynamics, displaying rich behaviours from bistability to oscillations and even chaos (Elowitz & Leibier, 2000). Furthermore, typical operating conditions for engineered circuits (bioreactors, gut microbiomes, ...=) are time varying, leading to more complex behaviours from even simple genetic circuits. To develop models of genetic circuit dynamics we require kinetic gene expression data generated at the test phase (Nuñez et al., 2017). This data must be integrated with models to enable characterization of parts and systems, as well as to metadata including the DNA sequence to enable automated design.

Thus there is a need for software design tools that integrate dynamical models, kinetic data, and DNA sequence via common exchange standards in a user-friendly and accessible fashion. Here we present LOICA (Logical Operators for Integrated Cell Algorithms), a Python package that simplifies dynamical genetic network modelling and design. LOICA is based on object-oriented programming and represents the complete operation of genetic circuits including their components, host cell and growth conditions with a set of classes. This approach means that the structure of a LOICA program is easy to understand and to customize, making a direct connection from computer code to genetic code and experimental protocols. LOICA provides a framework for freestyle genetic network/circuit design seamlessly integrating with experimental data registries (Guillermo et al., 2020) and registries of DNA designs via SBOL (McLaughlin et al., 2018).Our system integrates DNA sequence, mathematical models, and experimental data to enable automated characterization and prediction of circuit dynamics.

Methods

LOICA is a Python package built using object-oriented programming, and said objects interact with each other in order to generate models (https://github.com/SynBioUC/LOICA). The basic objects in the code are Operator and Regulator, and the interaction between them generates ordinary differential equations (ODE) models for gene expression as shown in Figure 1. The mathematical derivation of our model assumes steady state of mRNA given that their temporal dynamics are faster than protein dynamics (Yáñez Feliú, Vidal, Muñoz Silva, & Rudge, 2020).

The workflow in LOICA can be divided in two steps. Step one: GeneticNetwork construction, that means that we create a GeneticNetwork, add and wire its components.

- **Regulator:** Gene product that regulates gene expression, common Operator input. Ex. Transcription factor, tetR.
- **Producer:** Gene product that produces chemicals. Ex: Enzime, luxI.
- **Reporter:** Gene product that emits a signal. Ex: Green Fluorescent Protein (GFP)

- **Operator:** DNA fragment that encodes a transcriptional unit. It maps input to output through a Hill function where the parameter a is the basal expression, b is the regulated expression, K is the half max expression input concentration and n is the cooperativity degree. Can encapsulate Regulator, Producer, Reporter and Supplement.
- **GeneticNetwork:** DNA fragment that encodes a set of transcriptional units that conform a synthetic regulatory network, equivalent to a genetic circuit. Coordinates and encapsulates Operator, Regulator, Producer and Reporter interactions. Ex: Repressilator.

Step two: Assay setup, here we include context for the GeneticNetwork and set the experimental design.

- **Metabolism:** Here we add cellular context for gene expression in terms of biomass and growth rate is added.

- **Sample:** Encapsulates GeneticNetwork and Metabolism. Incorporates environmental information such as Supplements or chemicals, strain and media. Ex: 1 well in a plate, single cell.
- **Assay** Encapsulates a set of Samples. Runs and stores Samples temporal dynamics. Generates synthetic data. Can upload data to Flapjack. Ex: an experiment, 96 well plate.

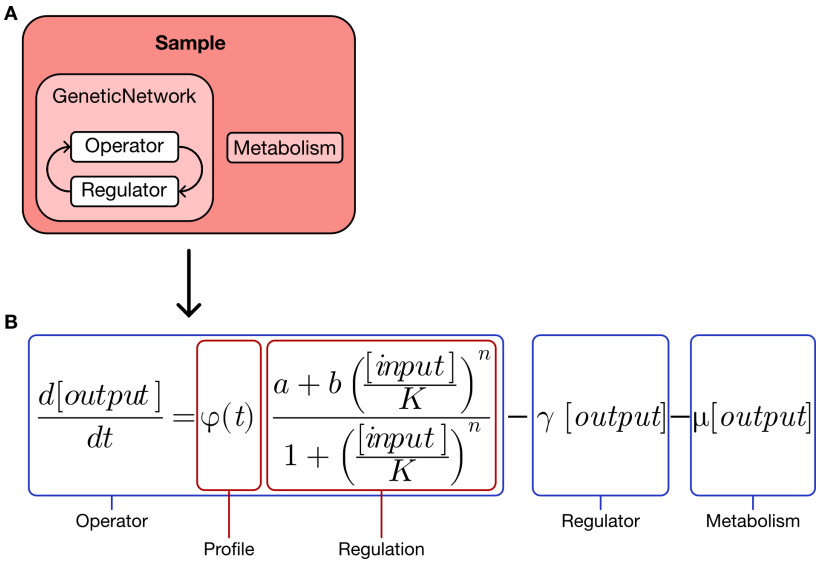


Figure 1. Diagram of model generation in LOICA. **A**. Diagram of a Sample encapsulating Metabolism and GeneticNetwork in which the Operator and Regulator are interacting to generate a model. **B**. Mathematical model generated through LOICA objects interaction. Here the Operator has a profile and a response or Hill function that states its regulation. Regulators can be inputs and outputs of Operators and incorporate a degradation rate. Metabolism incorporates dilution by growth.

LOICA were connected to FlapJack, a data management and analysis tool for genetic circuit characterization .To established a two-way communication we used identifier numbers (IDs) to link LOICA objects to FlapJack's objects through FlapJack's Python package (https://github.com/SynBioUC/flapjack) that interface FlapJack's REST API (https://github.com/SynBioUC/flapjack_api). This close connection allows straight access to the database and the use of all FlapJack's features including the webapp (https://github.com/SynBioUC/flapjack_frontend). The two-way communication is established between LOICA and FlapJack objects Figure 2. GeneticNetwork incorporates Vector ID and Reporter incorporates Signal ID to connect the models with the database Vector which is connected to the SBOL compatible design repository SynBioHub through uniform resource identifiers (URI) (https://github.com/SynBioHub/synbiohub).

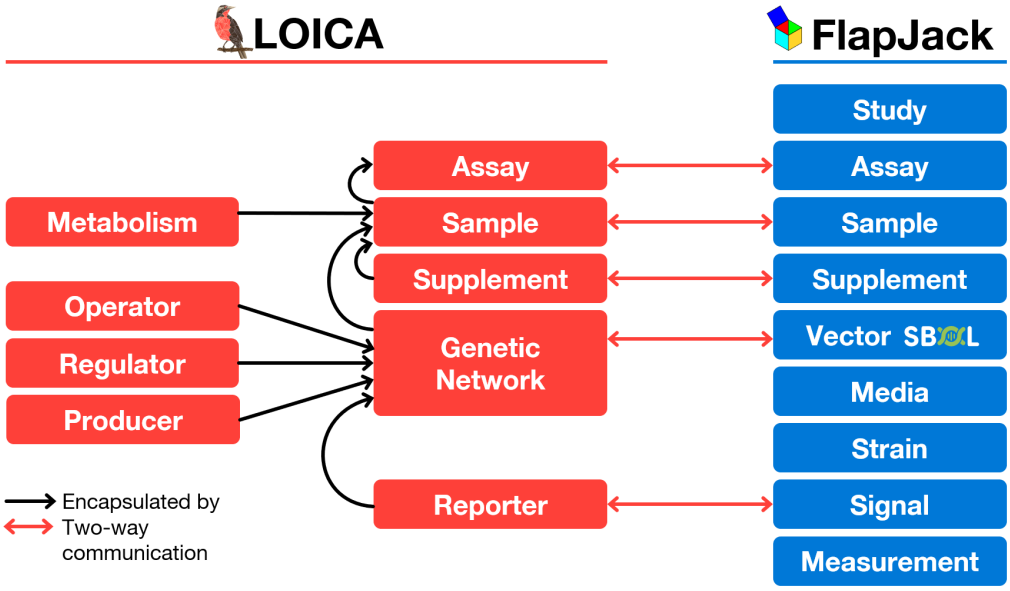


Figure 2. LOICA objects interaction and connection to FlapJack. Red and blue boxes correspond to different classes in LOICA and FlapJack respectively. Black arrows represent encapsulation, for example Operator and GeneProduct are part of GeneticNetwork. Red arrows represent two-way communication between LOICA and FlapJack classes. Supplement, Sample and Assay have a FlapJack counterpart. GeneticNetwork incorporates Vector ID and Reporter incorporates Signal ID to connect the models with the database.

Results

We developed LOICA (Logical Operators for Integrated Cell Algorithms) a Python package to model genetic circuits dynamics. It is two-way communicated with FlapJack, a data management and analysis tool for genetic circuit characterization. This communication provides straight access to data and design sequence in SBOL that enables characterization and simulation in the same tool.

We developed and implemented a new gene expression profile characterization method based on inverse problems in LOICA Figure 3A. This method is more accurate in average Figure 3B and is less prone to error at low biomass Figure 3C than linear inversion or direct method (Zulkower, Page, Ropers, Geiselmann, & De Jong, 2015). It

can be used also to characterize growth and to extract parameters of Hill functions.

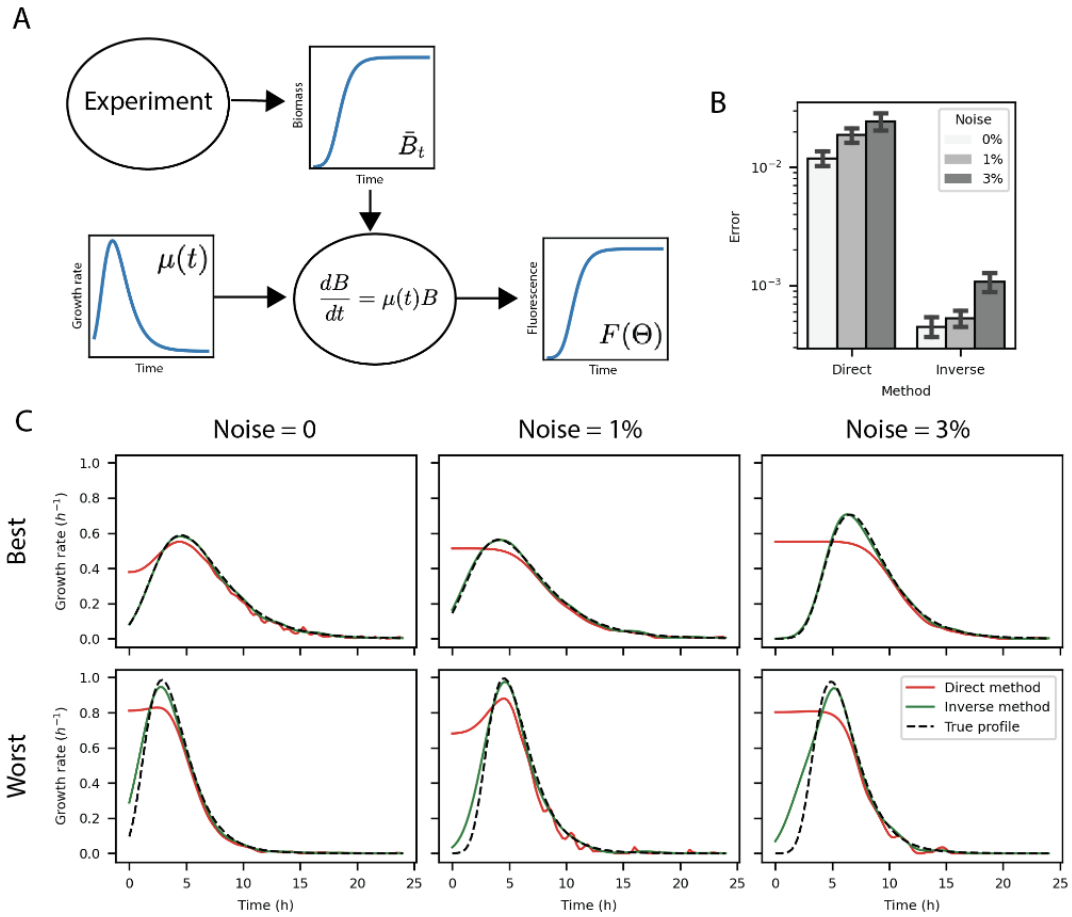
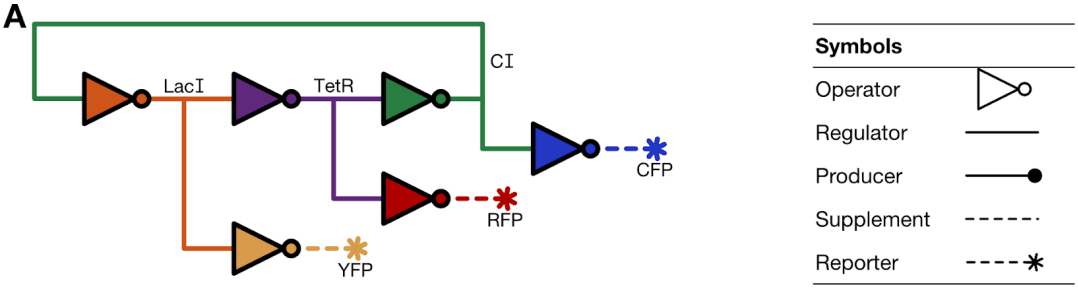


Figure 3. Comparison between the direct and inverse method to charactrize growth using 100 randomly parameterized Gompertz growth models. **A.** Diagram of the inverse method approach. **B.** Mean Squared Error between true and predicted profile using the direct and the inverse method. **C.** The inverse method is more robust to noise in early biomass measurements so that it correctly reconstructs the lag phase, which is missing from the linear inversion solution.

The repressilator is a convenient dynamic system study case (Gardner, Cantor, & Collins, 2000).To model it, we consider a simple balanced ring oscillator with three NOT Operators connected with three different Regulators. To link this circuit to a signal we add to the previous GeneticNetwork the same Operators using the same inputs but changing the outputs to three different Reporters, in this case fluorescent proteins, two-way communicated with FlapJack Signal Figure 4A. The code to generate this model is in Figure 4B and 4C. This feature is used to generate synthetic data from models that can be uploaded to the FlapJack database in one line of code to access to genetic circuit characterization tools, data management and data visualization.

A



B

Create GeneticNetwork

```
rep = GeneticNetwork(vector=vector.id[0])
```

Create and add Reporters

```
laci = Regulator(name='LacI', degradation_rate=1, init_concentration=5)
tetr = Regulator(name='TetR', degradation_rate=1)
ci = Regulator(name='cI', degradation_rate=1)
rep.add_regulators([laci, tetr, ci])
```

Create and add Reporters

```
cfp = Reporter(name='CFP', degradation_rate=1, signal_id=cfp.id[0])
yfp = Reporter(name='YFP', degradation_rate=1, signal_id=yfp.id[0])
rfp = Reporter(name='RFP', degradation_rate=1, signal_id=rfp.id[0])
rep.add_reporters([cfp, yfp, rfp])
```

Create, add, and wire Operators

```
ci_NOT_laci = Not(input=ci, output=laci, a=100, b=0, K=1, n=2)
laci_NOT_tetr = Not(input=laci, output=tetr, a=100, b=0, K=1, n=2)
tetr_NOT_ci = Not(input=tetr, output=ci, a=100, b=0, K=1, n=2)
rep.add_operators([ci_NOT_laci, laci_NOT_tetr, tetr_NOT_ci])
ci_NOT_cfp = Not(input=ci, output=cfp, a=100, b=0, K=1, n=2)
laci_NOT_yfp = Not(input=laci, output=yfp, a=100, b=0, K=1, n=2)
tetr_NOT_rfp = Not(input=tetr, output=rfp, a=100, b=0, K=1, n=2)
rep.add_operators([ci_NOT_cfp, laci_NOT_yfp, tetr_NOT_rfp])
```

C

Create Metabolism

```
metab = SimulatedMetabolism(biomass, growth_rate)
```

Create Samples

```
sample = Sample(circuit=rep,
                metabolism=metab,
                media=media.id[0],
                strain=strain.id[0]
               )
```

Create Assay

```
assay = Assay([sample],
              n_measurements=100,
              interval=0.25,
              name='Loica repressilator',
              description='Simulated repressilator generated by loica',
              biomass_signal_id=biomass_signal.id[0]
             )
assay.run()
```

Upload data to FlapJack

```
assay.upload(fj, study.id[0])
```

D

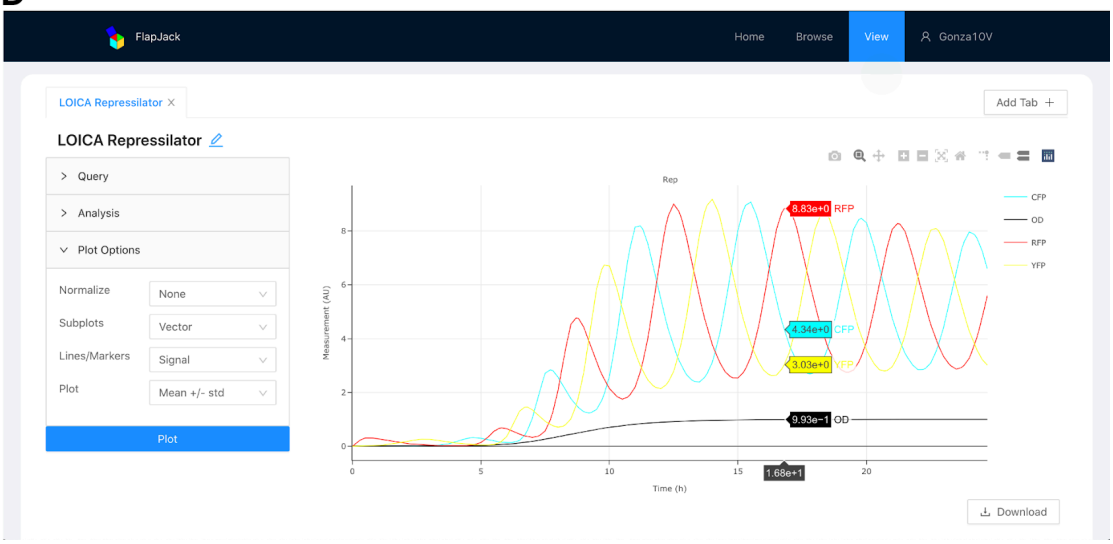


Figure 4. Example of a repressilator creation. **A.** LOICA diagram of the modeled repressilator circuit. **B.** GeneticNetwork construction, first step to create a LOICA model to design a repressilator. **C.** Assay setup, second step to create a LOICA model to design a repressilator. **D.** Data visualization via FlapJack. To obtain this image in

FlapJack click view, in query click studies and search Biodesign LOICA repressilator testing, the query will autocomplete downstream. In plot options click subplots and select vector to not show subplots. Click plot to visualize the data.

Conclusion

LOICA integrates genetic circuit dynamics model and design into Python using object-oriented programming to make it easy to understand for synthetic biologists and engineers with basic programming knowledge. It's capable of simulating repressilator dynamics and NOR gate steady state characterization data.

LOICA is designed to be easy to understand for new users and easy to customize for advanced users, where new characterized parts are easy to incorporate into LOICA using child classes. We propose a workflow that integrates LOICA, FlapJack and SBOL to accelerate the DBTL cycle in synthetic biology. An example of the workflow could be as follows: in the first design stage users can design three repressible transcription units (TU) or NOT gates that express GFP using SBOL tools such as pySBOL3, SBOLDesigner or SBOLCanvas. In the build stage you assemble those TU using Golden Gate for example. In the test stage users can make a dynamic induction curve in a plate reader measuring GFP fluorescence and optic density (OD) to upload that data into FlapJack. In the learn stage users can use all FlapJack visualization and analysis tools. This data can also be used by LOICA to characterize the TUs as NOT Operators. With LOICA data driven models with mathematical constraints you can design repressilators and look for likely functional designs initiating a new DBTL cycle. LOICA and FlapJack closes and will accelerate the DBTL cycle in synthetic biology.

Funding

GV was supported by a scholarship from the Institute for Biological and Medical Engineering, Pontificia Universidad Católica de Chile. GYF was supported by Beca Ayudante Doctorando scholarship from the Department of Chemical and Bioprocess Engineering, Pontificia Universidad Católica de Chile. TJR, GV, GYF, were supported by ANID PIA Anillo ACT192015.

References

Elowitz, M. B., & Leibier, S. (2000). A synthetic oscillatory network of transcriptional regulators. *Nature*, *403*(6767), 335–338. https://doi.org/10.1038/35002125

Gardner, T. S., Cantor, C. R., & Collins, J. J. (2000). Construction of a genetic toggle switch in Escherichia coli. *Nature*, *403*(6767), 339–342. https://doi.org/10.1038/35002131

Guillermo, Y., Earle, G., Berrocal, V. C., Mu, M., Nu, I. N., Matute, T. F., … Dahlin, J. (2020). *Flapjack : a data management and analysis tool for genetic circuit characterization*. 1–35.

McLaughlin, J. A., Beal, J., Mısırlı, G., Grünberg, R., Bartley, B. A., Scott-Brown, J., … Myers, C. J. (2020). The Synthetic Biology Open Language (SBOL) Version 3: Simplified Data Exchange for Bioengineering. *Frontiers in Bioengineering and Biotechnology*, *8*(September), 1–15. https://doi.org/10.3389/fbioe.2020.01009

McLaughlin, J. A., Myers, C. J., Zundel, Z., Mlslrll, G., Zhang, M., Ofiteru, I. D., … Wipat, A. (2018). SynBioHub: A Standards-Enabled Design Repository for Synthetic Biology. *ACS Synthetic Biology*, *7*(2), 682–688. https://doi.org/10.1021/acssynbio.7b00403

Nielsen, A. A. K., Ross, D., Der, B. S., Voigt, C. A., Densmore, D., Paralanov, V., … Strychalski, E. A. (2016). Genetic circuit design automation. *Science*, *352*(6281), aac7341–aac7341. https://doi.org/10.1126/science.aac7341

Nuñez, I., Matute, T., Herrera, R., Keymer, J., Marzullo, T., Rudge, T., & Federici, F. (2017). Low cost and open source multi-fluorescence imaging system for teaching and research in biology and bioengineering. *PLoS ONE*. https://doi.org/10.1371/journal.pone.0187163

Shin, J., Zhang, S., Der, B. S., Nielsen, A. A., & Voigt, C. A. (2020). Programming Escherichia coli to function as a digital display . *Molecular Systems Biology*, *16*(3), 1–12. https://doi.org/10.15252/msb.20199401

Yáñez Feliú, G., Vidal, G., Muñoz Silva, M., & Rudge, T. J. (2020). Novel Tunable Spatio-Temporal Patterns From a Simple Genetic Oscillator Circuit. *Frontiers in Bioengineering and Biotechnology*, *8*(August). https://doi.org/10.3389/fbioe.2020.00893

Zulkower, V., Page, M., Ropers, D., Geiselmann, J., & De Jong, H. (2015). Robust reconstruction of gene expression profiles from reporter gene data using linear inversion. *Bioinformatics*, *31*(12), i71–i79. https://doi.org/10.1093/bioinformatics/btv246