SimGenex: A System for Concisely Specifying Simulation of Biological Processes and Experimentation

Anyela Camargo
School of Computing Sciences
University of East Anglia
Norwich, Norfolk, UK
Email: a.camargo-rodriguez@uea.ac.uk

Jan T. Kim
School of Computing Sciences
University of East Anglia
Norwich, Norfolk, UK
Email: j.kim@uea.ac.uk

***Abstract*—Computational models enable advances in understanding essential features of living systems. Such models can be used to simulate data that can also be measured empirically. Generating such simulated data is frequently a key step in developing and validating models. However, precisely specifying a complex procedure of simulating data is notoriously difficult. The SimGenex language reported here is designed to simplify this task as it is applicable in research scenarios where several candidate models are considered, the mathematical details of regulatory interactions are only known partially or described semiquantitatively, the majority of kinetic parameters are not empirically measured, and a gene expression matrix is available as a basis of identifying the best model. SimGenex enables succinct and flexible descriptions of simulating the biological processes and experimental procedures that are the building blocks of most current wet lab experimental protocols. It enables specification of reproducibly executable workflows for validating computational models of biological systems, it facilitates pre-processing and transformation of data as it is frequently applied in gene expression data analysis and it provides support for comparing and discriminating alternative candidate models based on their ability to approximate the empirical dataset. The result of applying a SimGenex program to a computational model is a simulated dataset that can directly be compared to empirically measured *omic* data through the specification of a distance measure which can be used to discriminate the best model among a number of candidates.**

***Keywords*-Gene Regulatory Networks; Simulation; Systems Biology;**

I. Introduction

Gene expression measurements determine the amount of product of one or more genes in a biological sample. The amount or concentration of a gene product is called the *expression level* of the gene that encodes the product.

Samples for gene expression measurement are typically cultivated at controlled conditions. While the specific conditions depend on the object of research and the research question, the properties that are subject to control can generally be classified into *genetic properties* and *environmental conditions*. Genetic properties pertain to the genetic makeup of the subjects. Specifically, genes may be *knocked out* (loss of function mutations), or *overexpressed* (gain of function mutations). Typical environmental conditions applied in lab experimentation include treatment with agents such as hormones or drugs, variations in temperature, pH or salinity, and differences in supply of energy or nutrition.

Such signals are effectors impinging on cellular activities and on expression of some genes, or they result in the activation of such effectors. The affected genes frequently encode transcription factors which in turn alter expression of further genes. Perception of environmental signals can thus ripple through a cell's gene regulatory network (GRN), and ultimately change expression levels of many genes.

GRNs enable cells to react to environmental conditions in a genetically determined way. GRNs generate complex dynamics and patterns and attract much scientific interest, particularly since drug development and genetic engineering often involve targeted modification of GRN dynamics. GRN models offer a comprehensive understanding of disease progression, and they can help to predict clinical responses and can be vital to streamline efforts to identify the most promising candidates as early as possible in the drug development pipeline [1]. Therefore, computational GRN models are often used to predict GRN dynamics and to investigate the principles of GRN organisation.

The collection of expression levels of all genes in all samples is called an *expression set*, or, in recognition of the "genes $\times$ conditions" format of the set, an *expression matrix* $X = (x_{gc})$, where g indexes genes and c indexes conditions. The set of expression levels of a given gene g, measured in different samples, is called the *expression profile* (or profile, for short) of that gene, denoted by $\mathbf{x}_g$.

Gene expression measurements are obtained by wet lab methods such as rtPCR or microarrays. The readouts from these techniques are subjected to mathematical operations (e.g. for background correction, normalisation) to obtain estimates of gene expression levels. Gene expression data frequently contain artifacts that require some form of pre-processing (e.g. if a data set contains few negative expression values, this problem can be solved by adding a small offset to all values). After pre-processing, expression data typically is transformed into log-ratios [6], [2] by designating a reference condition c^* of expression levels (typically corresponding to the unperturbed wild type sample), and calculating $\log(x_{gc}/x_{gc^*})$ for all genes g and all columns c.

Gene expression profiles can be compared by choosing a *profile distance measure* d that quantifies how similar two profiles are. The distance measures currently supported are the Euclidean distance and the correlation distance [8]. The semi-metric correlation distance is defined as $1 - r(\mathbf{x}_g, \mathbf{x}_{g'})$, where $r(\mathbf{x}_g, \mathbf{x}_{g'})$ denotes the sample correlation coefficient between the expression profiles of genes g and g', and captures the similarity of "shape" of the profiles being compared [3].

A *simulated expression matrix* can be constructed by applying *in silico* operations to a suitable computational GRN model. Each individual operation reflects biological process or an experimental procedure. By comparing the results of such a simulation to empirical observations, GRN models can be systematically validated. Specifically, a simulated expression matrix Y can be compared to *target matrix* X of empirical gene expression levels by computing $\sum_g d(\mathbf{x}_g, \mathbf{y}_g)$. This *matrix distance* gives an indication of how well the GRN model captures the gene regulatory dynamics of the system from which the empirical measurements were taken, and it can be used to discriminate alternative computational GRN models.

Computational biology often requires reproducible performance of complex workflows to try to simulate the biological processes and experimental procedures that are the building blocks of most current wet lab experimental protocols. Performing *in silico* operations on GRN models typically requires programming in a general purpose language. For data analysis purposes, tools such as Taverna [7], designed to automatise bioinformatics analyses and EXACT [9], designed to represent biological laboratory protocols, have recently been developed. The SimGenex language defines a set of primary operations that are sufficiently general to simulate most standard experimental procedures. This is done within the transsys framework for GRN modelling [5], [4]. SimGenex specifications of operations that model experimentation are declarative and much shorter and simpler than equivalent simulations coded in a computer programming language. SimGenex also provides facilities for specifying mathematical transformations of the primary simulated expression values, and for specifying a distance measure for comparing matrices.

II. SimGenex Feature Overview

The core of a SimGenex program describes how to use a transsys GRN model to produce a simulated gene expression matrix. The `measurementmatrix` block describes how to transform the primary simulated matrix into a *measurement matrix* by e.g. computing log-ratios. Finally, the `discriminationsettings` block configures computation of the distance of the measurement matrix to a target matrix.

A. Simulating Gene Expression

The empirical data in the *target matrix* are normally produced by wet lab means such as rtPCR or microarray. It follows that a number of genotypes are exposed to a number of environmental conditions. In the simulated scenario, transsys GRN models represent genotypes. These are subjected to simulated conditions to produce simulated gene expression values that match the empirical scenario.

The columns of a matrix simulated by SimGenex are generated by creating an initial state and applying a sequence of primary simulation instructions to that state. The primary instructions provided by SimGenex are:

- `runtimesteps` to run a specified number of time steps,
- `knockout` to remove the specified gene from the transsys GRN model,
- `treatment` to set the expression level of a factor to a specified value,
- `overexpress` to insert a new, constitutively expressed gene into the GRN model,
- `setproduct` to alter the product encoded by a gene.

Instruction sequences that are used repeatedly can be declared as a `procedure`. Procedures may in turn invoke other procedures Thus, procedures can straightforwardly be reduced to sequences of primary instructions.

Columns in the simulated matrix are specified by `simexpression` declarations. Like procedures, simexpressions may be composed of primary instructions and procedure invocations. In addition, they also may contain `foreach` instructions. Such simexpressions define multiple columns in the simulated matrix. The `foreach` instruction enables very compact specifications of setups (e.g. when a number of strains are subjected to the same set of experimental conditions). For example, the declaration

```
simexpression s
{
  foreach: wildtype komutant;
  equilibration;
  foreach mock real;
  onehour;
}
```

specifies four columns in which the genotypes `wildtype` and `komutant` are subjected to `mock` and the `real` treatment. The procedures `komutant`, `mock` and `real` have to be defined in order for the above code fragment to work.

B. Computing the Simulated Matrix

In line with the wet lab scenario, the columns of a matrix simulated by SimGenex need to be transformed following the same protocols that were applied to compute the target matrix of empirical data. SimGenex uses the following blocks within the `measurementmatrix` section to specify such procedures:

- `measurementprocess`: specifies an `offset` parameter to normalise individual gene expression values and a `transformation` equation to indicate how expression values are transformed to simulate a column in a gene expression matrix, e.g. by a log-ratio transform.
- `measurementcolumns`: specifies the columns in the simulated expression matrix. Columns are computed by subjecting the expression levels in one or more simexpressions to mathematical operations, resulting in a column containing one value for each mapped factor of the candidate program. The idea is that the mathematical operations should be the same as those applied to the raw empirical data that have resulted in the empirical expression matrix (e.g. log-ratios where the ratio of a treatment to a control is calculated).

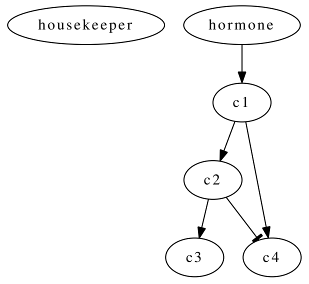
C. Discrimination Settings and Gene mapping

SimGenex allows the specification of a `distance` measure to compare the simulated matrix to a target matrix which can e.g. be used to discriminate the best GRN model from among a number of candidates. In addition SimGenex allows the specification of a mapping scheme, `genemapping`, whereby names of genes in the computational model can be mapped to names in the target matrix. These may e.g. be IDs designated by the microarray provider.

Beyond configuring matrix distance, the `discriminationsettings` section may provide further configuration to be used in the process of discriminating GRN models. Currently, SimGenex supports a `whitelist` of factors or genes which a discriminator may adjust. This feature is useful where parts of the GRN model are unknown, and the discriminator should therefore explore various alternatives for the unknown parts. As an example, where numerical parameters are unknown, these can be set by numerical optimisation.

III. Results

We demonstrate the use of SimGenex on the very simple regulatory network shown in Fig. 1, which is comprised of a constitutively expressed housekeeping gene and a cascade of four genes encoding factors `c1`, `c2`, `c3` and `c4`. The code of the SimGenex protocol to simulate measurements for the wildtype and all single gene knockout mutants is partially shown in Fig. 2. The complete code is posted on the transsys website [4]. Note that specifications for all columns in the `measurementcolumns` block and for all factors in the genemapping were not included. In line with standard practice, we use the expression levels in the wild type as the reference.



```
gene c1gene
{
  promoter
  {
    hormone: activate(0.01, 1.0);
  }
  product
  {
    default: c1;
  }
}

gene c2gene
{
  promoter
  {
    c1: activate(0.01, 1.0);
  }
  product
  {
    default: c2;
  }
}

gene c4gene
{
  promoter
  {
    constitutive: 0.1;
    c1: activate(0.01, 1.0);
    c2: repress(0.01, 1.0);
  }
  product
  {
    default: c4;
  }
}
```

Figure 1. Example network for demonstrating simulation of gene expression measurements with SimGenex. The graph shown at the top shows the overall network topology. The code below shows a part of the transsys model. The `housekeeper` is constitutively expressed and not subject to any regulation. The `hormone` activates `c1`, which is not expressed in the absence of `hormone`. Likewise, `c2` is not expressed in the absence of `c1`. In contrast to this, `c3` is expressed without `c1`, but `c2` increases its rate of expression and `c4` is activated by `c1` and repressed by `c4`.

```
procedure hormtreat
{ treatment: hormone = 1.0; }
procedure equilibration
{ runtimesteps: 100; }
procedure ko_c1
{ knockout: c1gene; }

simexpression all
{
  foreach: wt ko_hk
          ko_c1 ko_c2 ko_c3 ko_c4;
  equilibration;
  foreach: notreat hormtreat;
  treatmenttime;
}

measurementmatrix
{
  measurementprocess
  {
    offset: 0.1;
    transformation:
       log2(offset(x1)) - log2(offset(x2));
  }
}

measurementcolumns
{
  wt_notreat: x1 = all_wt_notreat,
    x2 = all_wt_notreat;
  kohk_notreat: x1 = all_ko_hk_notreat,
    x2 = all_wt_notreat;
  koc1_notreat: x1 = all_ko_c1_notreat,
    x2 = all_wt_notreat;
  wt_hormtreat: x1 = all_wt_hormtreat,
    x2 = all_wt_hormtreat;
  kohk_hormtreat: x1 = all_ko_hk_hormtreat,
    x2 = all_wt_hormtreat;
  koc1_hormtreat: x1 = all_ko_c1_hormtreat,
    x2 = all_wt_hormtreat;
}
}

discriminationsettings
{
  genemapping
  { factor housekeeper = "housekeeper";
     factor c1 = "c1";  }
  distance: correlation;
  whitelistdefs
  { factor: housekeeper c1 c2 c3 c4;
     gene: hkgene c1gene c2gene c3gene
       c4gene;}
}
```

Figure 2. Partial SimGenex code to simulate measurements for the wildtype and all single gene knockout mutants.

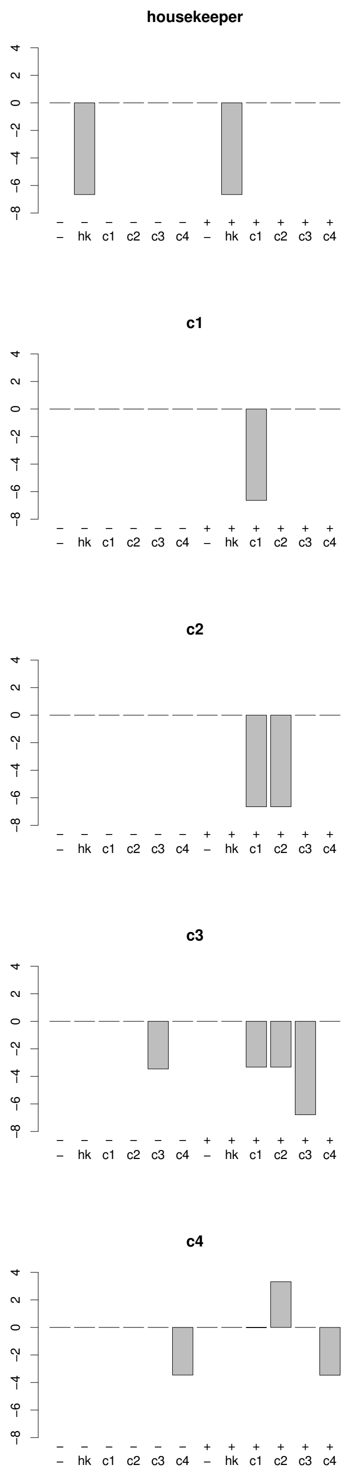


Figure 3. Simulated log-ratio gene expression profiles using the expression levels in the untreated wild type as the reference. The first five bars show expression measurement without, the second five bars show expression measurements with hormone treatment. For non-treated and treated samples, the wild type knockout mutants for all four genes are included.

Fig. 3 shows the simulated gene expression measurements. The two negative bars in the profile of `housekeeper` reflect the fact that this gene product's expression is abolished when the housekeeping gene is knocked out. Otherwise, the housekeeper's expression does not respond to any of the simulated conditions. Without `hormone` treatment, `c1gene` is not expressed, and as a consequence, `c2gene` is not expressed either (see Fig. 1). Therefore, knocking out these genes does not cause any changes in gene expression when no hormone treatment is applied. However, with hormone treatment, the genes in the cascade are expressed and as a consequence, knockouts have detectable effects on the downstream genes in the cascade.

IV. CONCLUSION

Computational biology often requires reproducible performance of complex workflows. For data analysis purposes, tools such as Taverna [7] and EXACT [9] have recently been developed. SimGenex complements these tools by enabling reproducible specification of simulations of biological processes and experimental procedures. The current main use of SimGenex is generating simulated matrices of expression values. Further, it facilitates specification of a distance measure to compare the simulated matrix to a target matrix comprised of gene expression data externally provided by wet lab means and provides support for discriminating the best gene regulatory network model from among a number of candidates. SimGenex is based on a small and generic set of operations that can be supported by many computational systems biology simulators, and provides new opportunities for unified description and comparison of computational models of living systems.

In our experience, most of the time required to execute a SimGenex program is typically used for simulation of gene expression dynamics. Therefore, significant speed-up can be achieved by re-using intermediate results where the instruction sequences of multiple columns in the simulated matrix share common prefixes, and the underlying GRN model is deterministic. In the near future we plan to optimise the SimGenex implementation accordingly. We also plan to set up a web service for accessing SimGenex and for assessing computational GRN models based on empirical target gene expression data.

ACKNOWLEDGEMENT

This work was supported by the Biotechnology and Biological Sciences Research Council, grant number BB/F009437/1.

REFERENCES

[1] D. K. Arrell and A Terzic. Network systems biology for drug discovery. *Nature*, 88, 2010.

[2] Patrik D'haeseleer. How does gene expression clustering work? *Nature Biotechnology*, 23:1499–1501, 2005.

[3] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown, and David Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, 95:14863–14868, 1998.

[4] Jan T. Kim *et.al.* The `transsys` home page, 2001-2010. `http://www.transsys.net/`.

[5] Jan T. Kim. `transsys`: A generic formalism for modelling regulatory networks in morphogenesis. In Jozef Kelemen and Petr Sosík, editors, *Advances in Artificial Life (ECAL 2001)*, volume 2159 of *Lecture Notes in Artificial Intelligence*, pages 242–251, Berlin Heidelberg, 2001. Springer Verlag.

[6] MAQC Consortium. The microarray quality control project shows inter- and intraplatform reproducibility of gene expression measurements. *Nature Biotechnology*, 24:1151–1161, 2006.

[7] Tom Oinn, Matthew Addis, Justin Ferris, Darren Marvin, Martin Senger, Mark Greenwood, Tim Carver, Kevin Glover, Matthew R. Pocock, Anil Wipat, and Peter Li. Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(17):3045–3054, 2004.

[8] John Quackenbush. Computational analysis of microarray data. *Nature Reviews Genetics*, 2:418–426, 2001.

[9] Larisa Soldatova, Wayne Aubrey, Ross D. King, and Amanda Clare. The EXACT description of biomedical protocols. *Bioinformatics*, 24:i295–i303, 2008.