CancerInSilico: An R/Bioconductor package for combining mathematical and statistical modeling to simulate time course bulk and single cell gene expression data in cancer

Thomas D Sherman[1], Luciane T Kagohara[1], Raymon Cao[1], Raymond Cheng[2], Matthew Satriano[3], Michael Considine[1], Gabriel Krigsfeld[1], Ruchira Ranaweera[4], Yong Tang[5], Sandra A Jablonski[6], Genevieve Stein-O'Brien[1,7], Daria A Gaykalova[8], Louis M Weiner[6], Christine H Chung[4], and Elana J Fertig[1,9,10]

[1] Department of Oncology, Division of Biostatistics and Bioinformatics, Sidney Kimmel Comprehensive Cancer Center, Johns Hopkins University, Baltimore, MD USA
[2] Science, Math and Computer Science Magnet Program, Poolesville High School, Poolesville, MD USA
[3] Department of Mathematics, University of Waterloo, Waterloo, Ontario, Canada
[4] Moffitt Cancer Center, Tampa, FL, USA
[5] Salubris Biotherapeutics, Inc, Gaithersburg, MD, USA
[6] Lombardi Comprehensive Cancer Center, Georgetown University, Washington, DC USA
[7] Institute of Genetic Medicine, Johns Hopkins University, Baltimore, MD USA
[8] Department of Otolaryngology-Head and Neck Surgery, Johns Hopkins University School of Medicine, Baltimore, MD USA
[9] Department of Applied Mathematics and Statistics, Johns Hopkins University
[10] Department of Biomedical Engineering, Johns Hopkins University

Contact tsherma4@jhu.edu, ejfertig@jhmi.edu

Abstract

Bioinformatics techniques to analyze time course bulk and single cell omics data are advancing. The absence of a known ground truth of the dynamics of molecular changes challenges benchmarking their performance on real data. Realistic simulated time-course datasets are essential to assess the performance of time course bioinformatics algorithms. We develop an R/Bioconductor package, *CancerInSilico*, to simulate bulk and single cell transcriptional data from a known ground truth obtained from mathematical models of cellular systems. This package contains a general R infrastructure for running cell-based models and simulating gene expression data based on the model states. We show how to use this package to simulate a gene expression data set and consequently benchmark analysis methods on this data set with a known ground truth. The package is freely available via Bioconductor: http://bioconductor.org/packages/CancerInSilico/

Introduction

Time course bioinformatics analysis techniques are emerging to delineate cellular composition and pathway activation from longitudinal genomics data [1,2]. However, benchmarking their performance is challenged by a lack of ground truth of the processes occurring in those datasets. For example, even relatively simple covariates, such as cellular density and proliferation rates impact experimental measures at a given time point, such as therapeutic sensitivity in cancer [3]. The interactions between these processes will introduce additional correlation structure between genes measured with genomics technologies. Simulated data can enable robust benchmarking of bioinformatics analysis methods for omics data. Statistical methods that utilize expected gene expression profiles from reference datasets to model the error distribution of bulk and single cell sequencing data are prominent [4–6]. Yet, there are few time course omics datasets to use as a benchmark and even fewer with known cellular-molecular dynamics. Therefore, new simulation systems with known ground truth are needed to benchmark the performance of emerging time course bioinformatics algorithms for bulk and single cell datasets.

Mathematical models of cellular dynamics are maturing in systems biology and can be used to track the state of the processes occurring in each cell in complex biological systems, such as cancer [7–13]. Some models simulate cell growth at a cellular level, where the population behavior is driven by the laws governing the individual cells and their interactions [14,15]. To further capture the complexity of biological systems, numerous multiscale and hybrid models linking cellular signaling to the equations of the cellular composition are emerging [16–18]. These models often require numerous parameters to simulate high throughput proteomic and transcriptional data and therefore often have similar complexity to real biological systems. Thus, mathematical models provide a robust framework from which to develop simulated time course datasets that are reflective of biological systems.

In this paper, we present a new software package to simulate time course transcriptional data. This is done by developing a general software framework to integrate mathematical models of cellular growth with statistical models of genomics data. The software is implemented in the R/Bioconductor package *CancerInSilico*. We simulate pathway activity based upon the simulated distribution of growth factor, state in the cell cycle, and cellular type. We couple a mathematical model from [14] with a statistical model from [19] to simulate transcriptional data based upon simulated pathway activity. We simulate data from microarrays and single cell RNA-seq using established platform-specific error distribution models [4,19,22]. Finally, we demonstrate how this framework can be used to benchmark time course analysis tools for genomics data.

Design and implementation

Software architecture

CancerInSilico is designed with an R user interface so that it is familiar to the bioinformatics community. The components of the simulation such as cell types and pathways are implemented as S4 classes in R. The cell model component of the simulation is implemented as an S4 class hierarchy, where features such as cell geometry (on-lattice vs off-lattice) form the basis for a set of models that individual implementations can inherit from. This allows different levels of the cell model to be considered separately so that the user can examine the effects of not just a single model but a whole class of models. The hierarchy also simplifies the number of parameters the user must interact with. Each level of the hierarchy contains its share of the overall parameters, so if the user wants to modify the low level implementation parameters, they can do so without worrying about any effects to the parameters upstream. This object oriented design simplifies the workflow by allowing each component of the model to be specified separately. In order to run the simulation, the user just needs to pass in any desired components along with a few high-level parameters.

All components of the simulation, including the cell model hierarchy, have a mirrored class structure in C++. While the classes in R contain the necessary parameters, the C++ classes also include the necessary routines to efficiently simulate a cell model. This architecture essentially allows the user to see a snapshot of the model in R and trust that the C++ backend will run it exactly as they prescribe. Moreover, this combines a simple user interface in R with a powerful, efficient backend in C++. The C++ library is exposed to R using the Rcpp package from CRAN.

The statistical model for gene expression simulation is written in R and exists outside of the previously mentioned class structure. It is intended to be an independent component that only needs the output of a cell simulation and a set of parameters in order to run. This way it is agnostic to any implementation details of the cell simulation, as well as any components that may be added to the cell simulation. The needed parameters are provided as an S4 class for convenience, allowing them to easily be saved alongside the simulation results.

Implementation details

The cellular growth simulation is driven by the cell model hierarchy and the peripheral components that can be added such as drugs and cell types. The `CellBasedModel` class at the top of the hierarchy specifies the relationship between a cell model and any peripheral components. It makes no requirements on the cell geometry or updating procedure of the model. In most cases, the user will be modifying parameters at this level. The model-component relationship is implemented with virtual functions in C++ so individual model implementations can override the default behavior. This top level class also uses pure virtual functions to specify the functions a cell model must implement.

We note that the `CellBasedModel` has an associated `Cell` class to separate cell-level logic with model-level logic. This is a design pattern seen across all levels of the hierarchy.

Any class that fully implements the specification of a `CellBasedModel` can be used in *CancerInSilico*, however it is convenient to define an intermediate class between the top-level `CellBasedModel` and an actual implementation. This layer describes a certain set of cell models, usually by specifying cell geometry, e.g. off-lattice vs on-lattice. This is a useful abstraction since the most computationally expensive aspects of a cell model often stem from the cell population data structure. If every model implementation required designing this data structure from scratch it would put an enormous burden on the developer. By having a pre-defined, efficient data storage and access API, new cell dynamics can be quickly prototyped and will come with an expected level of performance.

The intermediate layer specifies the cell geometry and structure, but the updating procedure must be handled by the actual cell model implementation. This lowest-level of the hierarchy is responsible for actually enforcing the desired cell mechanics. This is typically done by specifying some total energy function on the full cell population. Updating then involves several kinds of changes that are either accepted or rejected with a probability based on the total energy function. Most cell models in the literature [12, 14] can be identified by how they handle this updating procedure. By isolating this layer within the *CancerInSilico* architecture, such models can be easily implemented.

While the cellular growth simulation is an important part of *CancerInSilico*, the main feature of the package is the gene expression simulation. The connection between this simulation and the cellular growth simulation happens through user defined pathways that are then associated with gene expression changes in a corresponding set of genes. We define an intermediate variable (*P*) that is a continuous value between zero and one that records how active each biological pathway is within each modeled cell. The value of *P* in a given cell at a given time is determined by a user defined function in R. This allows for many types of pathways to be considered and for a great deal of expressiveness in how each pathway behaves. *CancerInSilico* also comes with some pre-defined pathways so that the burden is not entirely on the user to design the pathway behavior. Along with this user defined function, each pathway also has a set of genes annotated to it. Each gene in this set has a pre-specified expression range (*Gmin* to *Gmax*), determined either from a reference dataset or according to a specified distribution.

The function `inSilicoGeneExpression` combines the results of a call to `inSilicoCellModel` and a list of user defined pathways to simulate the requested type of transcriptional data. The first step is to create a matrix of mean expression values. This is done by evaluating each gene according to the specified behavior from the list of pathways. The expected expression value for each gene and each sample is given as:

$$g_{i,j} = \frac{1}{N}\sum_{k=1}^{N} P_k G_{max}^{k} + (1 - P_k) G_{min}^{k}, \quad (1)$$

where i indexes each gene, j each sample, and k each pathway. We note that while the mean used in eq (1) is provided as the default, *CancerInSilico* allows for user defined functions to combine pathway specific expression values. In bulk data, P_k is determined by computing the average value for pathway activity in a random set of N sampled cells, whereas in single cell data the value of P_k for each of the N sampled cells is used directly. The simulated gene expression value is obtained using a platform specific measurement error based on this expectation. A normal error model is used to simulate log transformed microarray data and a negative binomial error model, adapted from the code for LIMMA voom [22], is used to simulate bulk RNA-sequencing data. Measurement error for simulated single cell RNA-sequencing data are generated using the error and drop out models from Splatter [4].

Results

The *CancerInSilico* workflow

Running a cell simulation with *inSilicoCellModel*. The first step when simulating gene expression data with *CancerInSilico* is to create a cell simulation to serve as a reference point. This cell simulation will represent the underlying cellular processes driving the gene expression profiles. In order to run a cell simulation, we must call `inSilicoCellModel`. The three required arguments to this function are the initial number of cells in the simulation, the number of hours to simulate, and the initial density of the cell population. Optionally, it is possible to select the underlying mathematical model. A full description of the optional parameters is included in the supplemental material. An example call to the function might look like:

```
> cellModel = inSilicoCellModel(100, 72, 0.01, "DrasdoHohme")
```

Defining pathways. Before we can move on to simulating gene expression data, it is necessary to define the pathways which link the cell model state to the activity among a set of genes. *CancerInSilico* comes with a set of default pathways, however we can also explicitly define new pathways. In order to create a new pathway we must specify the names of the genes in the pathway and activity function which takes a cell model as an argument and returns a value between zero and one based on how active the pathway is at the current time point. Once a pathway is defined it must be calibrated either to a real data set or using a statistical distribution. This calibration step is important so that the range of the gene expression values is reasonable. Here is an example of calibrating a default pathway with a distribution. The mean expression levels for all genes is exponentially distributed and the range of expression values per gene is normally distributed.

```
> data(samplePathways) # load pwyMitosis, pwySPhase

> pwyMitosis = calibratePathway(pwyMitosis, lambda=20, stddev=2)
> pwySPhase = calibratePathway(pwySPhase, lambda=20, stddev=2)

> pathways = list(pwyMitosis, pwySPhase)
```

Simulating gene expression data with *inSilicoGeneExpression*. Now that we have a defined set of pathways and a completed cell simulation, we are able to simulate a gene expression data set. This step involves computing the mean level of expression in all the pathway genes and applying a statistical error model based on the type of data being generated. There are a few parameters that control this part of the simulation. `sampleFreq` and `nCells` specify how often samples are drawn and how many cells are in each sample. `RNAseq` and `singleCell` are Boolean parameters that specify the type of data to generate. A full description of the parameters can be found in the supplemental material. An example call to the function might look like:

```
> params = new("GeneExpressionParams", nCells=50, RNAseq=TRUE)

> exp = inSilicoGeneExpression(cellModel, pathways, params)
```

Example: Simulating time-course bulk data

Using the workflow described in the previous section, we simulate a microarray data set across 43 time points. The underlying mathematical model for cellular growth in this case is an off-lattice, cell-center model from Drasdo and Höhme [14]. We model pathways related to the phase transition from G to S and G to M, as well as a pathway related to contact inhibition. For the G to M and G to S pathways, the pathway activity is either zero or one at the current time point depending on whether or not the cell is transitioning phases. The contact inhibition pathway activity is defined by the "local density" of the cell, which is the proportion of surrounding area of a cell that is occupied by other cells. We run the cell model for 168 hours, enough for the cell population growth to slow down due to the density of the cells, and simulate gene expression for 150 genes. We generate a heatmap of the data using the `heatmap.2` function in R **(Figure 2a)**. We also run PCA on the resulting microarray data set and show that, as expected, time and cell phase are the processes driving the simulated gene expression **(Figure 2bc)**.

Example: Simulating time-course single-cell data

CancerInSilico also encodes an option to simulate single cell RNA-sequencing data to generate omics data that reflects the heterogeneity of the sample population. To model this heterogeneity, *CancerInSilico* allows us to label each cell as being from a distinct cell type. We have control over the distribution of cell-cycle lengths within each cell type through a user defined function in R. We apply this framework to model two distinct cell types, one with a mean cell cycle length of 12 hours and standard deviation of 4 hours (type A) and one with a mean cell cycle length of 36 hours and standard deviation of 4 hours (type B). This simulation models the pathway activity and corresponding gene expression changes for each cell with a negative binomial error model and dropout model adapted from Splatter [4]. The model then randomly samples a pre-specified number of cells. We apply this technique to simulate single cell RNA-sequencing data from a simulation of a population equally distributed between the two types described above (**Fig 3**). Each cell type is labeled as a pathway with binary values for activity to activate a gene set that corresponds to cellular identity. In this simulated single-cell RNA-seq data, we observe strong separation between cell types (**Fig 3a**) and time (**Fig 3b**) and observe a mixture between cell cycle phases (**Fig 3c**).

Example: Benchmarking analysis tools

We can use the wide range of simulation conditions in *CancerInSilico* to benchmark time course gene expression analysis methods. Moreover, by starting with biological parameters as opposed to statistical parameters, we can benchmark analysis methods on conditions we actually care about. *CancerInSilico* is particularly useful when the cellular processes underlying the simulation of interest have complex relationships with each other. This complexity is handled by the underlying cellular growth simulation, and while it does not perfectly capture the dependence between cellular processes, it does provide a standardized, justifiable method. Furthermore, *CancerInSilico* allows for the underlying cellular growth model to be easily swapped out so the benchmark itself can be tested for sensitivity to a particular model.

Here we explore the effects of dependent cellular processes when using Independent Component Analysis (ICA), implemented in the R package `fastICA,` to analyze our simulated RNA-seq data set. We define two cell-types which have identical properties and provide an associated pathway for each one. We also define a third pathway which is proportional to the growth rate of a cell. We have two datasets, one in which all pathways have a distinct set of genes, and one where the growth pathway contains all of the genes in each cell type specific pathway. In this way, the growth pathway is confounding the cell type specific pathways. We can see that ICA with two components perfectly separates the cells (**Fig 4a**) and temporal dynamics (**Fig 4b**) of the simulated data with no confounding. When the overlapping pathway is introduced some of the cells can no longer be separated by type (**Fig 4c**) although the temporal dynamics are still separable via ICA (**Fig 4d**). Thus, this provides one example of the utility of *CancerInSilico* to benchmark the sensitivity of a time course analysis algorithm to its underlying mathematical assumptions. Additional parameters may be varied and further cell cycle pathways introduced to the data to increase the complexity of these simulations and more closely mirror the complexity of the analysis tasks in real, time course genomics data.

Availability and future directions

We develop a new R/Bioconductor package *CancerInSilico* that couples mathematical models of cellular growth with statistical models of technical noise. Using this coupling to model changes in gene sets annotated to cell signaling pathways [4,19,22] enables simulation of time course bulk omics data. The modeling of individual cells in this system also enables simulation of time course, single cell RNA-sequencing data. *CancerInSilico* provides a wide range of parameter spaces for the user to explore when simulating time-course gene expression data and the modular design makes it possible to swap different cell models in and out of an existing simulation. Thus, this package provides software that can be used to benchmark the performance of methods for time course bioinformatics analysis from a known ground truth that is lacking in real data. We note that, to our knowledge, this is the first software package designed to simulate time course gene expression data.

The statistical models used to simulate gene expression data from pathway activity in *CancerInSilico* mirror the process by which omics tools estimate that activity. Namely, pathways are assumed to activate discrete sets of genes annotated to a common function based upon the modeled cellular state. Default parameters for the model yield omics profiles with strong separation between signaling pathways that are greatly simplified relative to those observed in real data. Therefore, applying omics algorithms to these default simulations may result in under estimation of the accuracy of their performance for real time course data. Future work is essential to benchmark the performance of the CancerInSilico algorithm to real data. Mathematical models of gene regulatory networks have been developed to model the dynamics of regulatory networks that lead to transcriptional changes [28,29]. Hybrid, multi-scale approaches that combine these network-based models with the cellular-scale models more accurately model the complexity of system-wide dynamics [16–18] and are a promising area for future work to simulate time course omics data. However, the complexity of these gene regulatory models and extensive parameterization will limit the straightforward validation of omics algorithms that is possible from the simplified statistical models employed in *CancerInSilico*. We note that the complexity of the time course data simulated with *CancerInSilico* can be tuned by modifying the overlap between genes in simulated pathways, altering cell type specific cell growth parameters, or increasing the variation in parameter values across cells. Thus, we recommend benchmarking time-course omics data analysis algorithms on simulated data generated from a wide range of these parameter values to fully assess their performance.

CancerInSilico is available as an R package on Bioconductor bioconductor.org/packages/CancerInSilico/ and the source code is made available at github.com/FertigLab/CancerInSilico. A live tutorial (vignette) is provided in the R package and link to a pre-rendered version is available in the GitHub README. *CancerInSilico* is supported and tested on Windows, Mac, and Linux. The source code to generate the figures seen in this paper can be found at github.com/FertigLab/CancerInSilico-Figures.

ACKNOWLEDGEMENTS

We thank Ludmila V Danilova, Alexander V Favorov, Emily Flam, Dylan Kelley, Feilim Mac Gabhann, Cristian Tomasetti, and members of NewPISlack for critical comments and feedback on this project.

FUNDING

This work was supported by NIH Grants CA177669, CA006973, CA212007, CA50633, CA51008, DE017982, and SPORE DE019032. This work was also supported by The

Cleveland Foundation Helen Masenhimer Fellowship, Johns Hopkins University Catalyst and Discovery Grants, and Johns Hopkins School of Medicine Synergy Award. This project has been made possible in part by grant number 2018-183444 from the Chan Zuckerberg Initiative DAF, an advised fund of Silicon Valley Community Foundation.

REFERENCES

1. Liang Y, Kelemen A. Dynamic modeling and network approaches for omics time course data: overview of computational approaches and applications. Brief. Bioinform. 2017;
2. Liang Y, Kelemen A. Computational dynamic approaches for temporal omics data with applications to systems medicine. BioData Min. 2017; 10:
3. Hafner M, Niepel M, Chung M, et al. Growth rate inhibition metrics correct for confounders in measuring sensitivity to cancer drugs. Nat. Methods 2016; 13:521–527
4. Zappia L, Phipson B, Oshlack A. Splatter: simulation of single-cell RNA sequencing data. Genome Biol. 2017; 18:
5. Frazee AC, Jaffe AE, Langmead B, et al. Polyester: simulating RNA-seq datasets with differential transcript expression. Bioinformatics 2015; 31:2778–2784
6. Griebel T, Zacher B, Ribeca P, et al. Modelling and simulating generic RNA-Seq experiments with the flux simulator. Nucleic Acids Res. 2012; 40:10073–10083
7. Barish S, Ochs MF, Sontag ED, et al. Evaluating optimal therapy robustness by virtual expansion of a sample population, with a case study in cancer immunotherapy. Proc. Natl. Acad. Sci. 2017; 114:E6277–E6286
8. Tran PT, Bendapudi PK, Lin HJ, et al. Survival and Death Signals Can Predict Tumor Response to Therapy After Oncogene Inactivation. Sci. Transl. Med. 2011; 3:103ra99-103ra99
9. Chmielecki J, Foo J, Oxnard GR, et al. Optimization of Dosing for EGFR-Mutant Non-Small Cell Lung Cancer with Evolutionary Cancer Modeling. Sci. Transl. Med. 2011; 3:90ra59-90ra59
10. Picco N, Sahai E, Maini PK, et al. Integrating Models to Quantify Environment-Mediated Drug Resistance. Cancer Res. 2017; 77:5409–5418
11. Rockne R, Alvord EC, Rockhill JK, et al. A mathematical model for brain tumor response to radiation therapy. J. Math. Biol. 2009; 58:561–578
12. Szabó A, Merks RMH. Cellular Potts Modeling of Tumor Growth, Tumor Invasion, and Tumor Evolution. Front. Oncol. 2013; 3:
13. Ghaffarizadeh A, Heiland R, Friedman SH, et al. PhysiCell: an Open Source Physics-Based Cell Simulator for 3-D Multicellular Systems. 2017;
14. Drasdo D, Höhme S. Individual-based approaches to birth and death in avascular tumors. Math. Comput. Model. 2003; 37:1163–1175
15. Gallaher J, Anderson A. The role of contact inhibition in intratumoral heterogeneity: An off-lattice individual based model. 2016;
16. Rejniak KA, Anderson ARA. Hybrid models of tumor growth. Wiley Interdiscip. Rev. Syst. Biol. Med. 2011; 3:115–125
17. Clancy CE, An G, Cannon WR, et al. Multiscale Modeling in the Clinic: Drug Design and Development. Ann. Biomed. Eng. 2016; 44:2591–2610

18. Yankeelov TE, An G, Saut O, et al. Multi-scale Modeling in Clinical Oncology: Opportunities and Barriers to Success. Ann. Biomed. Eng. 2016; 44:2626–2641
19. Fertig EJ, Favorov AV, Ochs MF. Identifying context-specific transcription factor targets from prior knowledge and gene expression data. IEEE Trans. Nanobioscience 2013; 12:142–149
20. Matys V, Fricke E, Geffers R, et al. TRANSFAC: transcriptional regulation, from patterns to profiles. Nucleic Acids Res. 2003; 31:374–378
21. Subramanian A, Tamayo P, Mootha VK, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc. Natl. Acad. Sci. U. S. A. 2005; 102:15545–15550
22. Law CW, Chen Y, Shi W, et al. voom: precision weights unlock linear model analysis tools for RNA-seq read counts. Genome Biol. 2014; 15:R29
23. Fertig EJ, Ren Q, Cheng H, et al. Gene expression signatures modulated by epidermal growth factor receptor activation and their relationship to cetuximab resistance in head and neck squamous cell carcinoma. BMC Genomics 2012; 13:160
24. Ritchie ME, Phipson B, Wu D, et al. limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res. 2015; 43:e47–e47
25. Patro R, Duggal G, Love MI, et al. Salmon provides fast and bias-aware quantification of transcript expression. Nat. Methods 2017; 14:417–419
26. Soneson C, Love MI, Robinson MD. Differential analyses for RNA-seq: transcript-level estimates improve gene-level inferences. F1000Research 2015; 4:1521
27. Love MI, Huber W, Anders S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. 2014; 15:
28. Chasman D, Fotuhi Siahpirani A, Roy S. Network-based approaches for analysis of complex biological systems. Curr. Opin. Biotechnol. 2016; 39:157–166
29. Trairatphisan P, Mizera A, Pang J, et al. Recent development and biomedical applications of probabilistic Boolean networks. Cell Commun. Signal. 2013; 11:46
30. Fertig EJ, Ozawa H, Thakar M, et al. CoGAPS matrix factorization algorithm identifies transcriptional changes in AP-2alpha target genes in feedback from therapeutic inhibition of the EGFR network. Oncotarget 2016; 5:
31. Stein-O'Brien G, Kagohara LT, Li S, et al. Integrated time course omics analysis distinguishes immediate therapeutic response from acquired resistance. 2018;
32. Kleyman M, Sefer E, Nicola T, et al. Selecting the most appropriate time points to profile in high-throughput studies. eLife 2017; 6:

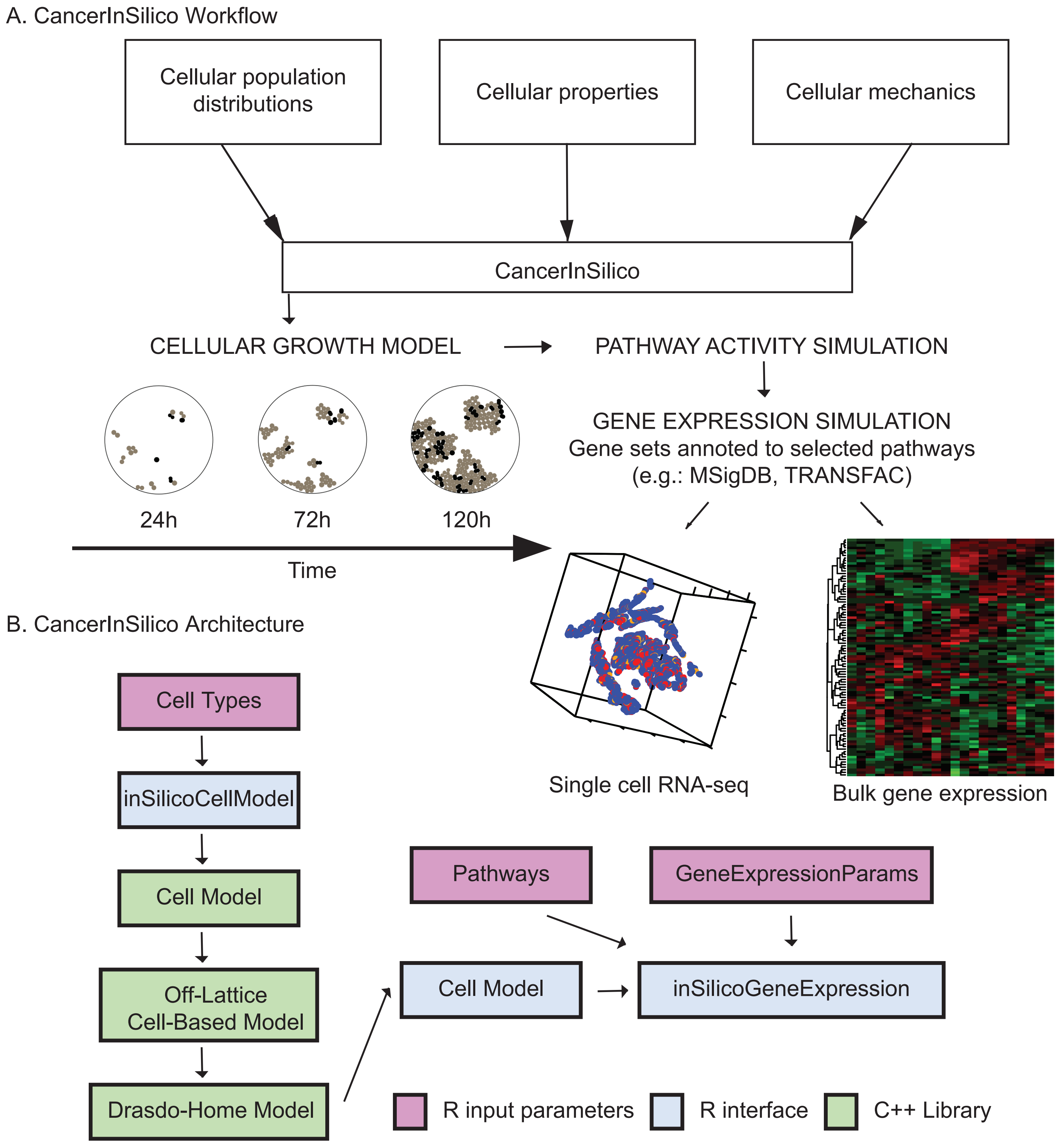
LIST OF FIGURES

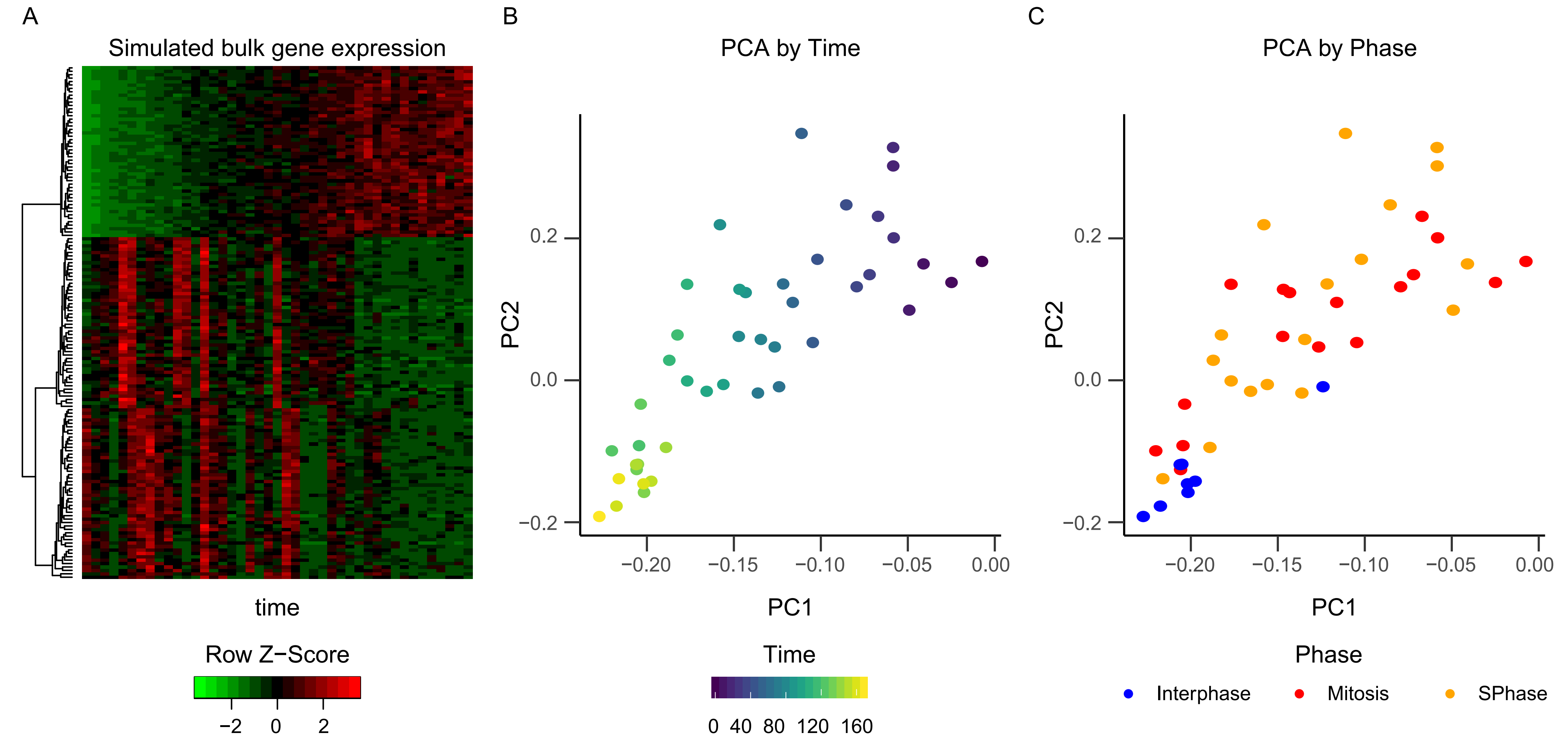
Fig 1. Visual representation of CancerInSilico. *(a) Overview of association between mathematical and statistical modeling to generate simulated time course gene expression data. (b) Each component of the software is shown with arrows indicating how it is related to the rest of the components. R input arguments shown in pink, R software components shown in blue, and C++ components shown in green.*

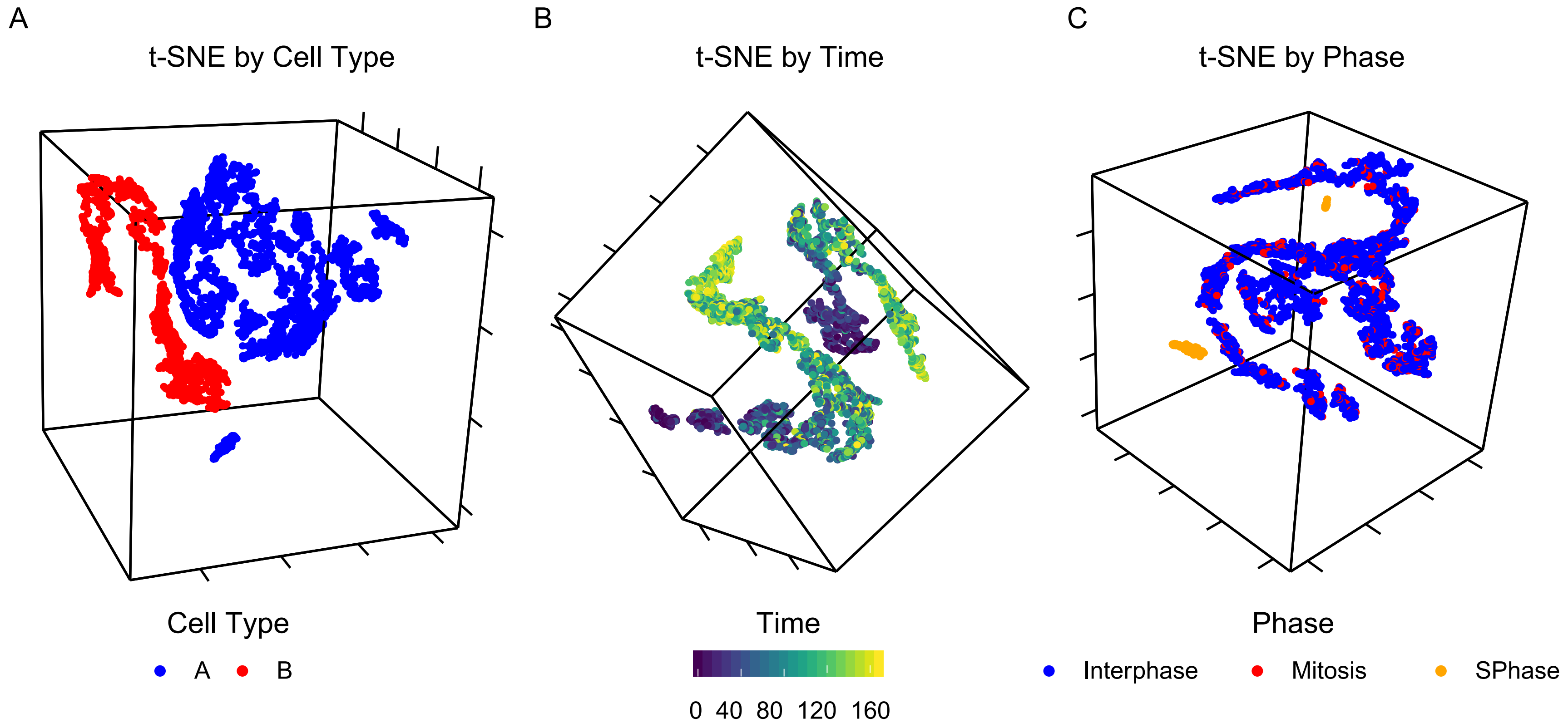
Fig 2. PCA of simulated time course microarray data. *(a) Heatmap of microarray data. Plot of first two principal components colored by (b) time and (c) cell phase.*

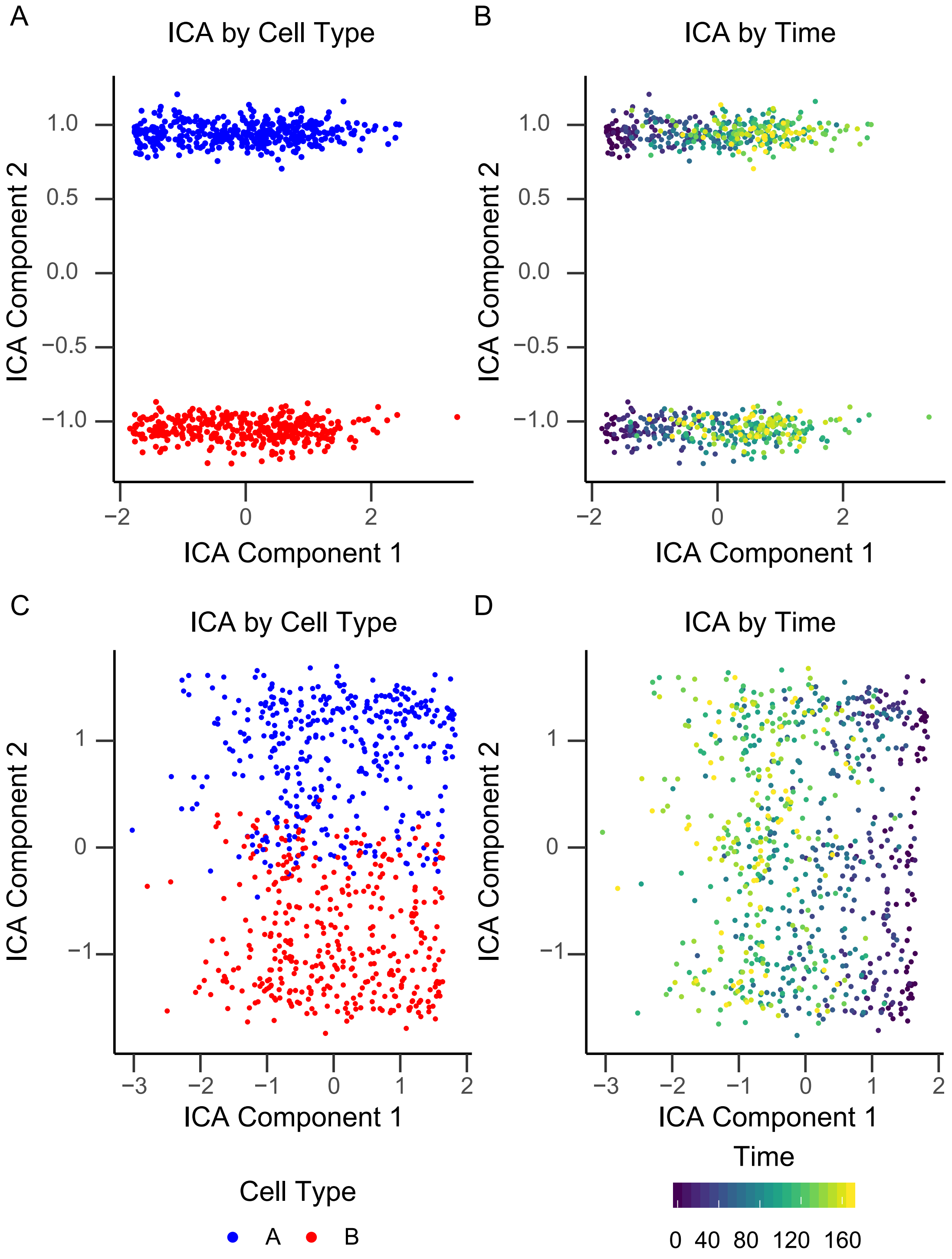
Fig 3. T-SNE of simulated time course single cell RNA-sequencing data for a population with cells of types A and B. *Points colored by (a) cell type, (b) time, and (c) cell cycle phase.*

Fig 4. Benchmarking ICA when cell type pathways are confounded with third pathway. *ICA with no confounding colored by (a) cell type and (b) time. ICA with third pathway confounding colored by (c) cell type and (d) time.*


A. CancerInSilico Workflow
Cellular population distributions
Cellular properties
Cellular mechanics
CancerInSilico
CELLULAR GROWTH MODEL
PATHWAY ACTIVITY SIMULATION
GENE EXPRESSION SIMULATION
Gene sets annoted to selected pathways
(e.g.: MSigDB, TRANSFAC)
24h
72h
120h
Time
Single cell RNA-seq
Bulk gene expression
B. CancerInSilico Architecture
Cell Types
inSilicoCellModel
Cell Model
Off-Lattice Cell-Based Model
Drasdo-Home Model
Pathways
GeneExpressionParams
Cell Model
inSilicoGeneExpression
R input parameters
R interface
C++ Library


A
Simulated bulk gene expression
time
Row Z−Score
−2
0
2
B
PCA by Time
PC2
0.2
0.0
−0.2
−0.20
−0.15
−0.10
−0.05
0.00
PC1
Time
0
40
80
120
160
C
PCA by Phase
PC2
0.2
0.0
−0.2
−0.20
−0.15
−0.10
−0.05
0.00
PC1
Phase
Interphase
Mitosis
SPhase


A
t-SNE by Cell Type
Cell Type
● A ● B
B
t-SNE by Time
Time
0 40 80 120 160
C
t-SNE by Phase
Phase
● Interphase ● Mitosis ● SPhase


A
ICA by Cell Type
B
ICA by Time
ICA Component 2
ICA Component 1
1.0
0.5
0.0
−0.5
−1.0
−2
0
2
C
ICA by Cell Type
D
ICA by Time
1
0
−1
−3
−2
−1
0
1
2
Cell Type
● A ● B
Time
0 40 80 120 160