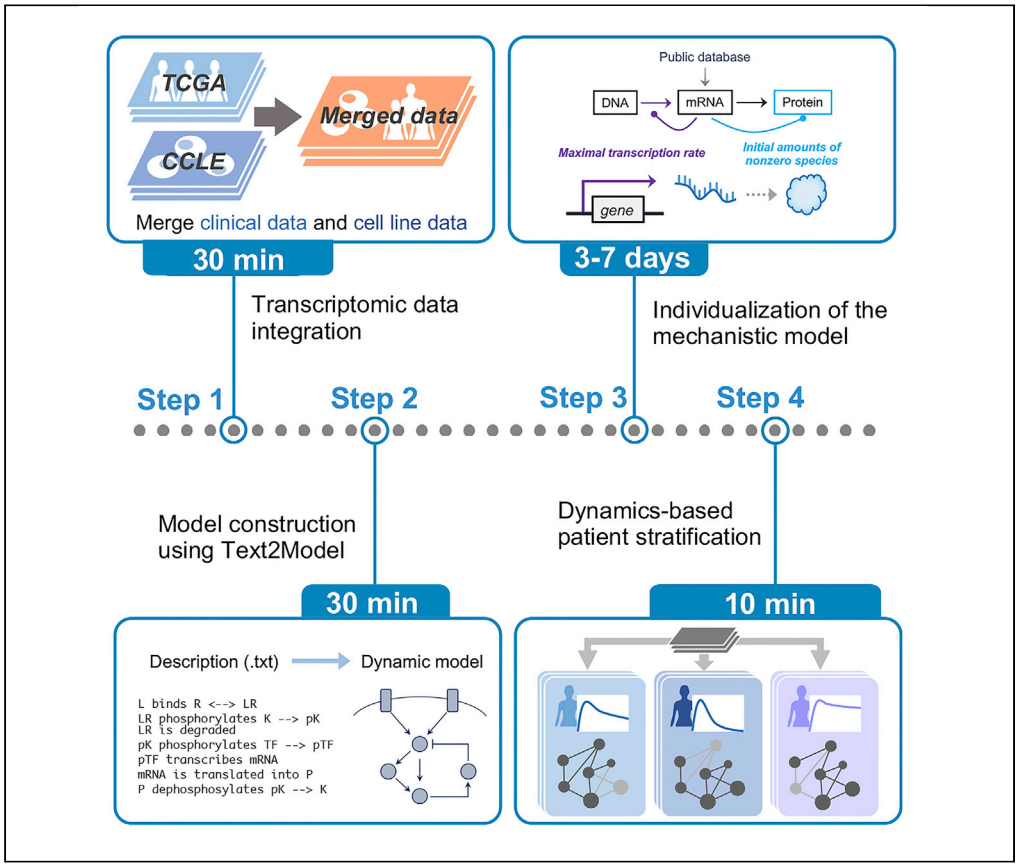
STAR Protocols

CellPress
OPEN ACCESS

Protocol

Protocol for stratification of triple-negative breast cancer patients using *in silico* signaling dynamics



Personalized kinetic models can predict potential biomarkers and drug targets. Here, we provide a step-by-step approach for building an executable mathematical model from text and integrating transcriptomic datasets. We additionally describe the steps to personalize the mechanistic model and to stratify patients with triple-negative breast cancer (TNBC) based on *in silico* signaling dynamics. This protocol can also be applied to any signaling pathway for patient-specific modeling.

Publisher's note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.

Hiroaki Imoto, Sawa Yamashiro, Ken Murakami, Mariko Okada

himoto@protein.osaka-u.ac.jp (H.I.)
mokada@protein.osaka-u.ac.jp (M.O.)

Highlights

A computational framework for patient-specific modeling

Integration of clinical data and cell line data for model calibration

Building a mechanistic dynamic model from .txt file

Stratification of patients with breast cancer based on *in silico* signaling dynamics

Imoto et al., STAR Protocols 3, 101619
September 16, 2022 © 2022 The Author(s).
https://doi.org/10.1016/j.xpro.2022.101619

STAR Protocols

CellPress
OPEN ACCESS

Protocol

Protocol for stratification of triple-negative breast cancer patients using *in silico* signaling dynamics

Hiroaki Imoto,[1,3,4,*] Sawa Yamashiro,[1,3] Ken Murakami,[1] and Mariko Okada[1,2,5,*]

[1]Institute for Protein Research, Osaka University, Suita, Osaka 565-0871, Japan

[2]Center for Drug Design and Research, National Institutes of Biomedical Innovation, Health and Nutrition, Ibaraki, Osaka 567-0085, Japan

[3]These authors contributed equally

[4]Technical contact

[5]Lead contact

*Correspondence: himoto@protein.osaka-u.ac.jp (H.I.), mokada@protein.osaka-u.ac.jp (M.O.)
https://doi.org/10.1016/j.xpro.2022.101619

SUMMARY

Personalized kinetic models can predict potential biomarkers and drug targets. Here, we provide a step-by-step approach for building an executable mathematical model from text and integrating transcriptomic datasets. We additionally describe the steps to personalize the mechanistic model and to stratify patients with triple-negative breast cancer (TNBC) based on *in silico* signaling dynamics. This protocol can also be applied to any signaling pathway for patient-specific modeling.

For complete details on the use and execution of this protocol, please refer to Imoto et al. (2022).

BEFORE YOU BEGIN

The temporal activation dynamics of signaling pathways play important roles for cell fate decisions (Purvis and Lahav, 2013). Therefore, we hypothesized that signaling dynamics can be further utilized as prognostic biomarkers for human diseases. However, the majority of available data obtained from patients represent static snapshots taken at a single point in time, and not time-resolved dynamics. To overcome this problem, we developed a Patient-Specific Modeling in Python (Pasmopy), an open-source package for the development of dynamic pathway models that are individualized to patient-specific data (Imoto et al., 2022). Using this tool, users can generate personalized mechanistic models from The Cancer Genome Atlas (TCGA) transcriptomic data and time-course training data for signaling activity obtained from cell lines, perform patient-specific simulations, and stratify patients based on *in silico* signaling dynamics. This protocol describes in detail the step-by-step method for model construction of the ErbB signaling network, parameterization of the models, integration of transcriptomic data, and stratification of breast cancer patients based on signaling dynamics. This model-based stratification can be applied to any signaling pathway and other types of cancer by replacing the input model descriptions as well as the training and clinical datasets. Overall, the personalized kinetic modeling approach presented herein can facilitate the identification of key molecular mechanisms in individual patients, which would otherwise be difficult by solely analyzing correlations between individual gene signatures and clinical outcomes.

Manual installation of required package components

Timing: 10 min

1. Clone the package and move to the breast_cancer/ directory:

This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

```
>git clone https://github.com/pasmopy/breast_cancer.git

>cd breast_cancer
```

Note: This repository contains all the codes for transcriptomic data integration, model construction, patient-specific simulations, and model-based stratification.

2. Install all Python packages required for the patient-specific simulations:

```
>python -m pip install -r requirements.txt
```

3. Install all R packages required for the transcriptomic data analysis:

```
$ R

> install.packages("dplyr")

> install.packages("edgeR")

> install.packages("sva")

> install.packages("tibble")

> install.packages("data.table")

> install.packages("stringr")

> install.packages("BiocManager")

> BiocManager::install("BioinformaticsFMRP/TCGAbiolinksGUI.data")

> BiocManager::install("BioinformaticsFMRP/TCGAbiolinks")

> install.packages("biomaRt")

> install.packages("cluster")

> install.packages("ComplexHeatmap")

> install.packages("circlize")

> install.packages("viridisLite")
```

All the required R packages above can be installed via executing "install_requirements.R".

4. (Optional) Install `BioMASS.jl` for parameter estimation:

```
>julia

>using Pkg; Pkg.add("BioMASS")
```

Testing execution environment for patient-specific modeling

Timing: 1–4 h (depending on the number of available CPU cores)

5. Install and run `pytest`:

```
>python -m pip install pytest

>pytest
```

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Software and algorithms		
Python 3.7.2	Python Software Foundation	https://www.python.org
breast_cancer v0.1.0	(Imoto et al., 2022)	https://doi.org/10.5281/ZENODO.6781265
pasmopy v0.1.0	(Imoto et al., 2022)	https://github.com/pasmopy/pasmopy
biomass v0.5.2	(Imoto et al., 2020)	https://github.com/biomass-dev/biomass
numpy v1.19.2	(Van Der Walt et al., 2011)	https://numpy.org
scipy v1.6.2	(Virtanen et al., 2020)	https://scipy.org
pandas v1.2.4	pandas – Python Data Analysis Library	https://pandas.pydata.org
seaborn v0.11.2	(Waskom, 2021)	https://seaborn.pydata.org
Julia 1.6.2	The Julia Programming Language	https://julialang.org
BioMASS.jl v0.5.0	(Imoto et al., 2020)	https://github.com/biomass-dev/BioMASS.jl
R 4.0.2	The R Foundation	https://www.r-project.org
TCGAbiolinks v2.25.0	(Colaprico et al., 2016)	https://bioconductor.org/packages/TCGAbiolinks/
sva v3.38.0	Zhang et al. (2020)	https://bioconductor.org/packages/sva/
biomaRt v2.46.3	(Durinck et al., 2009)	https://bioconductor.org/packages/biomaRt/
edgeR v3.36.0	(Robinson et al., 2009)	https://bioconductor.org/packages/edgeR/
ComplexHeatmap v2.10.0	(Gu et al., 2016)	https://bioconductor.org/packages/ComplexHeatmap/
ComplexHeatmap v2.10.0	(Gu et al., 2016)	https://bioconductor.org/packages/ComplexHeatmap/
BiocManager v1.30.18	Bioconductor	https://www.bioconductor.org/install/
cluster v2.1.3	Rousseeuw et al., 2015	https://cran.r-project.org/web/packages/cluster/index.html
Circlize v0.4.15	(Gu et al., 2014)	https://github.com/jokergoo/circlize
data.table v1.14.2	Rdatatable	https://github.com/Rdatatable/data.table
viridisLite v0.4.0	https://rdrr.io/cran/viridis/	https://sjmgarnier.github.io/viridis/
tibble v3.1.7	tidyverse	https://tibble.tidyverse.org/
dplyr v1.0.9	tidyverse	https://dplyr.tidyverse.org/
stringr v1.4.0	tidyverse	https://stringr.tidyverse.org/

STEP-BY-STEP METHOD DETAILS

Transcriptomic data integration

Timing: 30 min

⚠ CRITICAL: Steps 1 through 8 require approximately 11 GB of memory space and 6.3 GB of hard disk space.

In this step, we first obtain transcriptomic data of cancer patients from TCGA database (Weinstein et al., 2013) and normalize the transcriptomic data to be utilized for personalized modeling. We also obtain transcriptomic data of cell lines from the Cancer Cell Line Encyclopedia (CCLE) (Barretina et al., 2012), which is used for parameter determination and for training of the ErbB network model (Imoto et al., 2022). Subsequently, the batch effect in these datasets is removed using ComBat-seq (Zhang et al., 2020) so they can be merged and handled equally as model inputs. The details of this process are as follows.

1. Start R.
   a. Navigate to transcriptomic_data/ and start R:

```
>cd transcriptomic_data

>R
```

b. Read integration.R:

```
>source("integration.R")
```

Note: The functions executed in this step are defined in this file.

2. Obtain clinical information from the TCGA breast cancer dataset for manual analysis.

```
>outputClinical("BRCA")
```

Note: With this function, all clinical information registered in the dataset (e.g., patient ID, age, sex, and race) can be obtained. Under the default settings, the information obtained in this step is not used in Pasmopy (the information required, such as prognosis, stage, and age will be extracted in a later step); however, this information can be used for manual analysis and to check the quality of the dataset. This function uses the "GDCquery clinic" function from TCGA biolinks. The clinical information of patients is saved in "BRCA_clinical.csv".

3. Get patient list and clinical information for TCGA data analysis.

```
>outputSubtype("BRCA")
```

Note: In this step, the clinical information required for model-based patient stratification using Pasmopy, such as patient ID, subtype, age, and prognosis, is obtained. All patients' information is obtained in this step (patients are selected in the next step). This function uses the "TCGAquery_subtype" function and saves subtype information of the TCGA dataset in "BRCA_subtype.csv". Pasmopy stores the output in a variable named "subtype," which is used for the analysis described below.

4. Patient Selection.

```
>patientSelection(type = subtype,
        ID = "patient",
        pathologic_stage %in% c("Stage_I", "Stage_II"),
        age_at_initial_pathologic_diagnosis < 60)
```

Note: This is the preparation step for retrieving transcriptomic data from the TCGA database. In this step, patients are selected for the analysis that is to follow. Patients are selected by their pathological stage and the age at which they were initially diagnosed, which is listed in the metadata acquired in the previous step (the metadata is stored in the "subtype" variable). In this analysis, stage I and stage II patients who were under 60 years of age at their initial pathological diagnosis were used. In the end, 419 patients were included in subsequent analyses.

Users can freely change the selection criteria according to their needs. If the program fails to retrieve transcriptomic data, verify that the attribute names have been typed exactly as they are in the metadata (e.g., "pathologic_stage", "age_at_initial_pathologic_diagnosis", "Stage_I", "Stage_II"). For example, "Stage1" or "StageI" will not work for the TCGA-BRCA analysis because they do not match the attribute names in "BRCA_subtype.csv".

The "Type" argument should be the output of the outputSubtype("BRCA") function (the default is "Type=subtype," do not change it). The "ID" argument is the name of the column that includes the patient ID (if you are using the TCGA-BRCA dataset, use " ID="patient" ").

5. Download transcriptomic data of breast cancer cells from TCGA.

```
>downloadTCGA(cancertype = "BRCA",
        sampletype = c("01", "06"),
        outputresult = FALSE)
```

Note: Next, transcriptomic data are downloaded from TCGA using the conditions set above. "Sampletype" defines which type of tissue is used. The sample type code used herein follow the TCGA coding scheme (https://gdc.cancer.gov/resources-tcga-users/tcga-code-tables/sample-type-codes). For example, if the argument is set to "01" and "06", the function will fetch data from "Primary Solid Tumor" and "Metastatic" samples.

Note: If "Outputresult = TRUE" is set, the patients' transcriptomic data will be saved as a .csv file. There is no need to save the data as a .csv file if you continue the following analysis (the default is "false"). The directory in which the file will be saved is the directory in which the code is run (transcriptomic_data).

6. Download transcriptomic data from CCLE.

```
>downloadCCLE(cancertype = "BREAST",
        outputresult = FALSE)
```

Note: Next, the transcriptomic data of the cell lines are also downloaded from the CCLE database for parameter estimation of the model. Transcriptomic data from four cell lines and time-course training data on signaling activity from the corresponding cell lines were used to estimate the model parameters for this study (described in step 12). The concept of parameter estimation is explained in a subsequent section. In this example code, the https://data.broadinstitute.org/ccle/CCLE_RNAseq_genes_counts_20180929.gct.gz dataset is downloaded. Data is selected by cancer type from this dataset.

Note: If "Outputresult = TRUE" is set, the CCLE transcriptomic data are saved as a .csv file. There is no need to save the data as a .csv file if you continue the following analysis (the default is "false"). The directory in which the file will be saved is the directory in which the code is run (transcriptomic_data).

7. Merge TCGA data and CCLE data.

```
>mergeTCGAandCCLE(outputresult = FALSE)
```

Note: Next, TCGA data and CCLE data are merged, but these two transcriptomic datasets have distinct biases in their expression values which are caused by the different experimental settings. This bias is called the "batch effect," which must be eliminated before merging the datasets. The code above executes ComBat-seq, which is a tool for removing batch effects in sequence data and merging different datasets consecutively. After calculation, the adjusted read counts of each sample are obtained and saved as "totalreadcounts.csv," which will be used in the next step. If "Outputresult = TRUE" is set, the counts after performing ComBat-seq will be saved as "merged_TCGA_CCLE.csv".

8. Normalize counts.

```
>normalization(min=40000000, max=140000000)
```

Note: Finally, the read counts are normalized. This normalization allows the comparison of gene expression data between patients. Transcripts per million (TPM) are calculated and relative log expression (RLE) normalization is conducted. In general, TPM normalization adjusts the bias of counts caused by gene length, and RLE normalization adjusts the bias of counts caused by the difference in the number of read counts of each sample. The edgeR is used for RLE normalization. Gene lengths are obtained from the Ensembl database using biomaRt. Samples are selected based on the number of reads within the range of "min" to "max". After the selection, data from 369 patients remained for further analysis. The output file is "TPM_RLE_postComBat.csv". This file is later used in step 11.

Construction of a comprehensive model of the ErbB signaling network

Timing: 30 min

This section describes the creation of a mechanistic model from text and the preparation for incorporating gene expression data into the model. pasmopy.Text2Model is a useful Python class for building an ordinary differential equation (ODE) model from a text file describing biochemical reactions. We used this method to build a mechanistic model of the ErbB signaling network (Figure 1). For

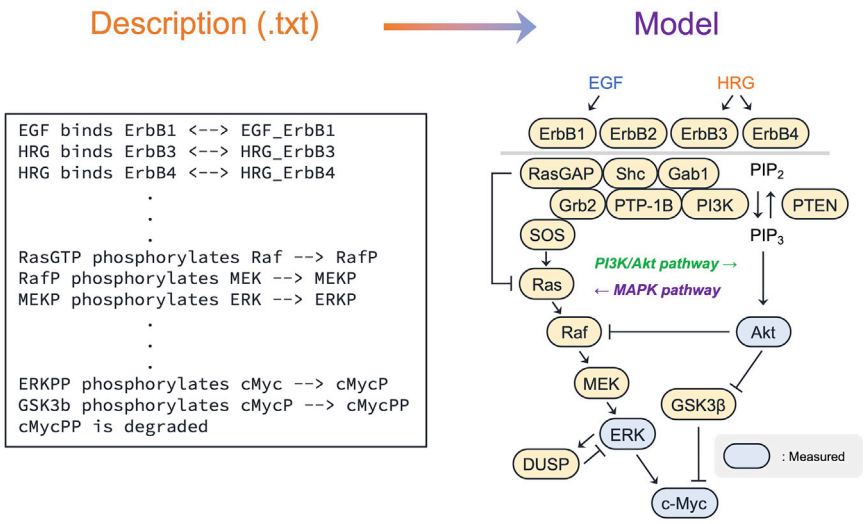


Figure 1. Text-to-model conversion using pasmpy.Text2Model

The text contains different types of biochemical events involved in the ErbB signaling, including binding, dissociation, phosphorylation, transcription, translation, and degradation, which can be automatically converted into kinetic equations. Reproduced with permission from iScience (Imoto et al., 2022).

further details of the model, please refer to the following papers: (Birtwistle et al., 2007; Imoto et al., 2022; Nakakuki et al., 2010).

9. Convert the text file into an executable model.

```
>import os

>from pasmopy import Text2Model

>Text2Model(os.path.join("models", "erbb_network.txt")).convert()
```

Note: pasmopy.Text2Model currently contains 14 reaction rules for gene regulation and biochemical reactions. Detailed options for this can be found in the online documentation: https://pasmopy.readthedocs.io.

10. Rename erbb_network/ to CCLE_name or TCGA_ID, e.g., MCF7_BREAST or TCGA_3-C_AALK_01A for individualization of the model.

```
>import shutil

>shutil.move(

  os.path.join("models", "erbb_network"),

  os.path.join("models", "breast", "TCGA_3C_AALK_01A")

)
```

11. Edit SearchParam class.

```
import os

import numpy as np

from pasmopy import Individualization

from . import __path__

from .name2idx import C, V

from .set_model import initial_values, param_values

incorporating_gene_expression_levels = Individualization(parameters=C.NAMES, spe-
cies=V.NAMES, transcriptomic_data=os.path.join("transcriptomic_data",

"TPM_RLE_postComBat_BRCA_BREAST.csv"), gene_expression={

"ErbB1": ["EGFR"], "ErbB2": ["ERBB2"], "ErbB3": ["ERBB3"], "ErbB4": ["ERBB4"], "Grb2":
["GRB2"], "Shc": ["SHC1", "SHC2", "SHC3", "SHC4"], "RasGAP": ["RASA1", "RASA2", "RASA3"],
"PI3K": ["PIK3CA", "PIK3CB", "PIK3CD", "PIK3CG"], "PTEN": ["PTEN"], "SOS": ["SOS1",
"SOS2"], "Gab1": ["GAB1"], "RasGDP": ["HRAS", "KRAS", "NRAS"], "Raf": ["ARAF", "BRAF",
"RAF1"], "MEK": ["MAP2K1", "MAP2K2"], "ERK": ["MAPK1", "MAPK3"], "Akt": ["AKT1", "AKT2"],
"PTP1B": ["PTPN1"], "GSK3b": ["GSK3B"], "DUSP": ["DUSP5", "DUSP6", "DUSP7"], "cMyc":
["MYC"]}, read_csv_kws={"index_col": "Description"})

class SearchParam(object):

...

  def update(self, indiv):

    x = param_values()
```

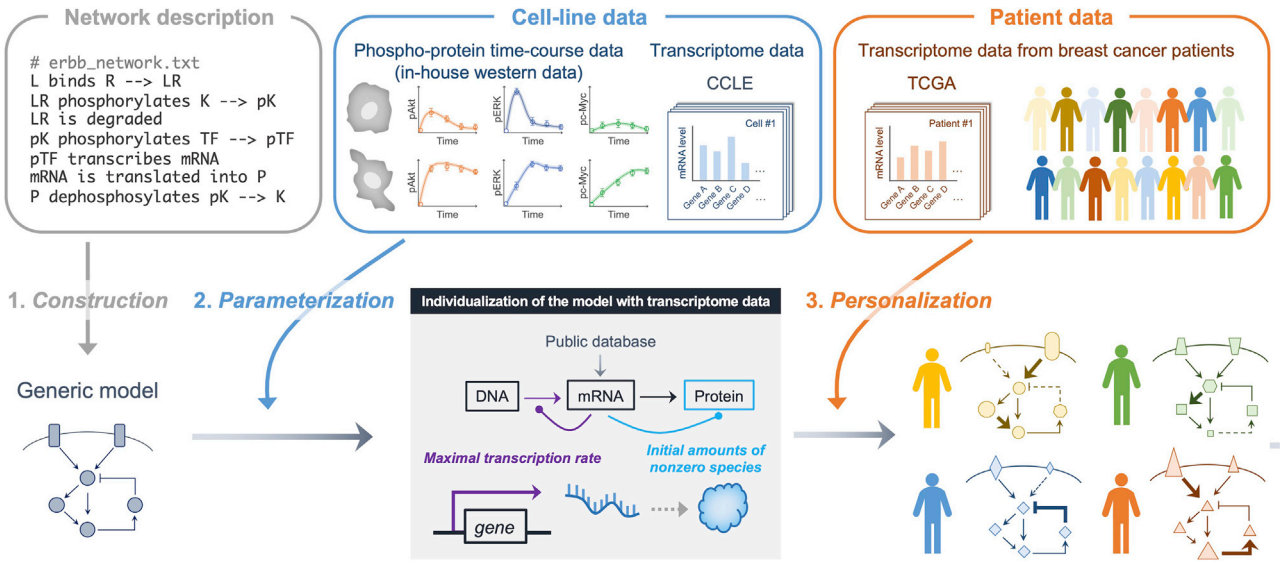


Figure 2. Preparation for the incorporation of gene expression data to parameterize and personalize the mechanistic model
(1) Building a generic model from text. (2) Determining model parameters using transcriptomic data provided in the CCLE database and phospho-protein time-course data. (3) Personalizing the mechanistic model using transcriptomic data provided in the TCGA database. Reproduced with permission from iScience (Imoto et al., 2022).

```
    y0 = initial_values()
    for i, j in enumerate(self.idx_params):
      x[j] = indiv[i]
    for i, j in enumerate(self.idx_initials):
      y0[j] = indiv[i + len(self.idx_params)]
    # As maximal transcription rate
    x[C.V291] = incorporating_gene_expression_levels.as_reaction_rate(
                __path__[0].split(os.sep)[-1], x, "V291", "DUSP")
    x[C.V310] = incorporating_gene_expression_levels.as_reaction_rate(
                __path__[0].split(os.sep)[-1], x, "V310", "cMyc")
    # As initial conditions
    y0 = incorporating_gene_expression_levels.as_initial_conditions(
            __path__[0].split(os.sep)[-1], x, y0)
...
```

Note: After editing SearchParam class as above, the model can incorporate gene expression data to estimate the maximal transcription rate and/or the initial amount of nonzero species when it is being parameterized and personalized (Figure 2).

Individualization of the mechanistic model

Timing: 3–7 days

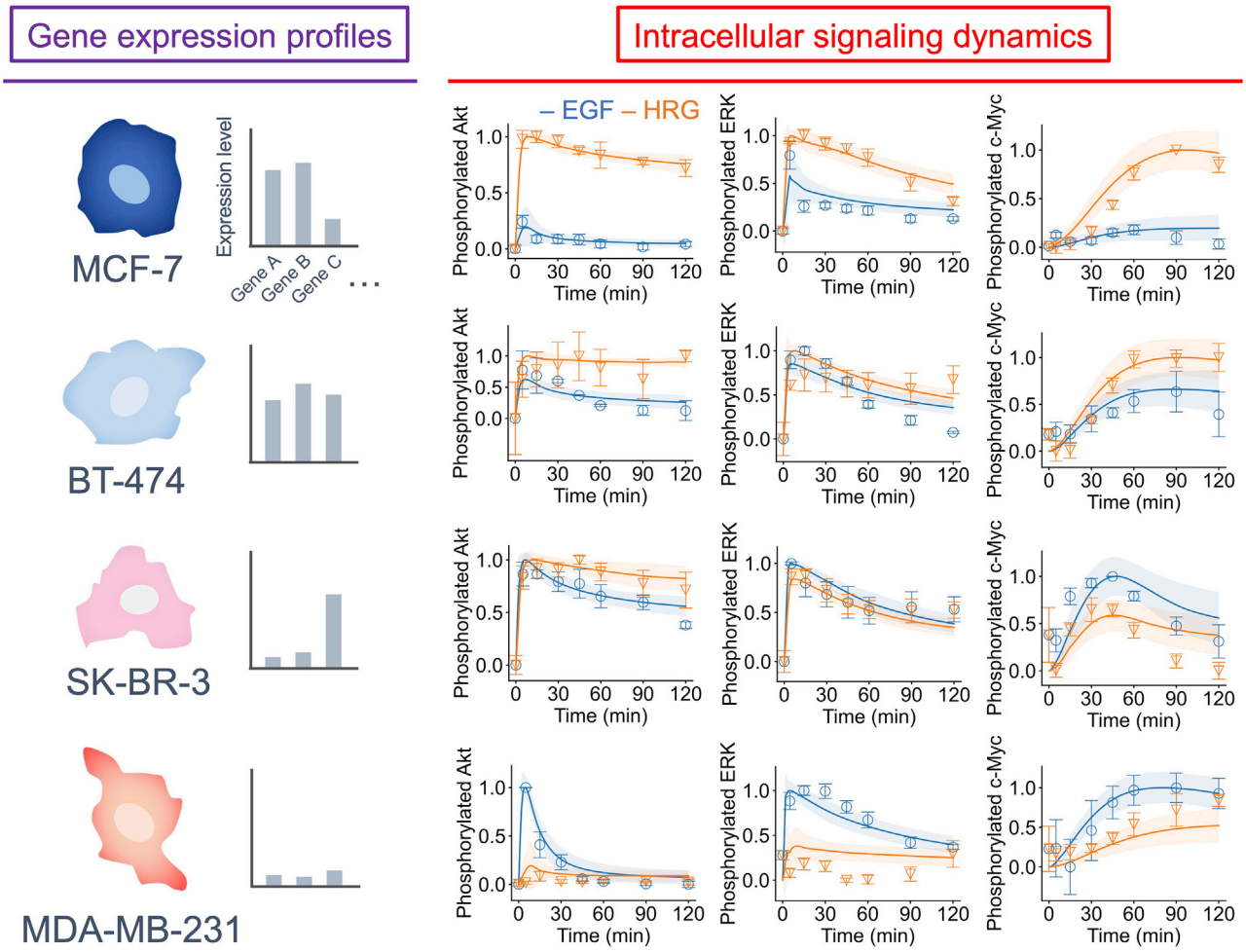


Figure 3. Parameter estimation using cell-line datasets

The model parameter was trained on time-series Akt, ERK, and c-Myc phosphorylation levels obtained from four breast cancer cell lines, namely MCF-7, BT-474, SK-BR-3, and MDA-MB-231, stimulated with EGF and HRG. The points (blue squares, EGF; orange triangles, HRG) denote experimental data, solid lines denote simulations, and shaded areas denote the SD. For all panels, error bars denote the SE for three independent experiments. Reproduced with permission from iScience (Imoto et al., 2022).

In this step, the parameters for the personalized kinetic models are determined, including the kinetic constants and weighting factors. It was assumed that the reaction parameters are unique to the molecular species involved in a reaction event and are identical across cell lines and patients. By training the quantitative relationship between expression values of the model species and the temporal patterns of intracellular signaling activities, the models can predict signaling dynamics in each patient upon adding the corresponding transcriptomic data as an input to the model. Phospho-protein time-course datasets obtained from four breast cancer cell lines (MCF-7, BT-474, SK-BR-3, and MDA-MB-231) stimulated with epidermal growth factor (EGF) and heregulin (HRG) were used to train the model parameters (Figure 3).

12. Build a mechanistic model to identify model parameters.

```
>import os

>from pasmopy import Text2Model

>Text2Model(os.path.join("models", "erbb_network.txt"), lang="julia").convert()
```

13. Run optimize_parallel.sh.

```
>mv erbb_network_jl training
>cd training
>mkdir errout
>sh optimize_parallel.sh # It will take more than a few days to optimize parameters.
>cd ..
```

Note: In this step, BioMASS.jl (Imoto et al., 2020) was used, but in most cases, the pasmopy.optimize() function can be used for parameter estimation.

14. Move optimization results to patient-specific models.

```
>julia
>using BioMASS
>param2biomass(``training'');
>exit()
>python
>import shutil
>breast_cancer_models = []
>path_to_models = os.path.join("models", "breast")
>for model in os.listdir(path_to_models):
  if os.path.isdir(os.path.join(path_to_models, model)) and (
    model.startswith("TCGA_") or model.endswith("_BREAST")
  ):
    breast_cancer_models.append(model)
# Set optimized parameters
>for model in breast_cancer_models:
  shutil.copytree(
    os.path.join("training", "erbb_network_jl", "dat2npy", "out"),
    os.path.join(path_to_models, f"{model}", "out"),
)
```

15. Run patient-specific simulations.

```
>import os
>import shutil
>from pathlib import Path
>from pasmopy import PatientModelSimulations
>import models.breast
>TCGA_ID = [
```

```
    l.strip() for l in Path("models", "breast", "sample_names.txt").read_text("utf-8").
splitlines()

]

# Create patient-specific models

>for patient in TCGA_ID:

    if patient != "TCGA_3C_AALK_01A":

      shutil.copytree(

        os.path.join("models", "breast", "TCGA_3C_AALK_01A"),

        os.path.join("models", "breast", f"{patient}"),

      )

# Execute patient-specific models

>simulations = PatientModelSimulations(models.breast.__package__, TCGA_ID)

>simulations.run()
```

Stratification of TNBC patients based on ErbB signaling dynamics

Timing: 10 min

In this study, *in silico* signaling dynamics which were generated from personalized kinetic models in the previous step were used to stratify breast cancer patients (Figure 4). In the example below, the maximum activation level is used as the dynamic characteristic for the classification of TNBC patients.

16. Run subtyping() function.

```
>simulations.subtyping(

  fname=None,

  dynamical_features={

      "Phosphorylated_Akt": {"EGF": ["max"], "HRG": ["max"]},

      "Phosphorylated_ERK": {"EGF": ["max"], "HRG": ["max"]},

      "Phosphorylated_c-Myc": {"EGF": ["max"], "HRG": ["max"]},

  }

)
```

Note: New dynamic characteristics, for example the species' droprate, can be added as follows:

```
>import numpy as np

> def get_droprate(time_course: np.ndarray) -> float:

    return - (time_course[-1] - np.max(time_course)) / (len(time_course) - np.argmax(time_
course))

    >simulations.response_characteristics["droprate"] = get_droprate
```

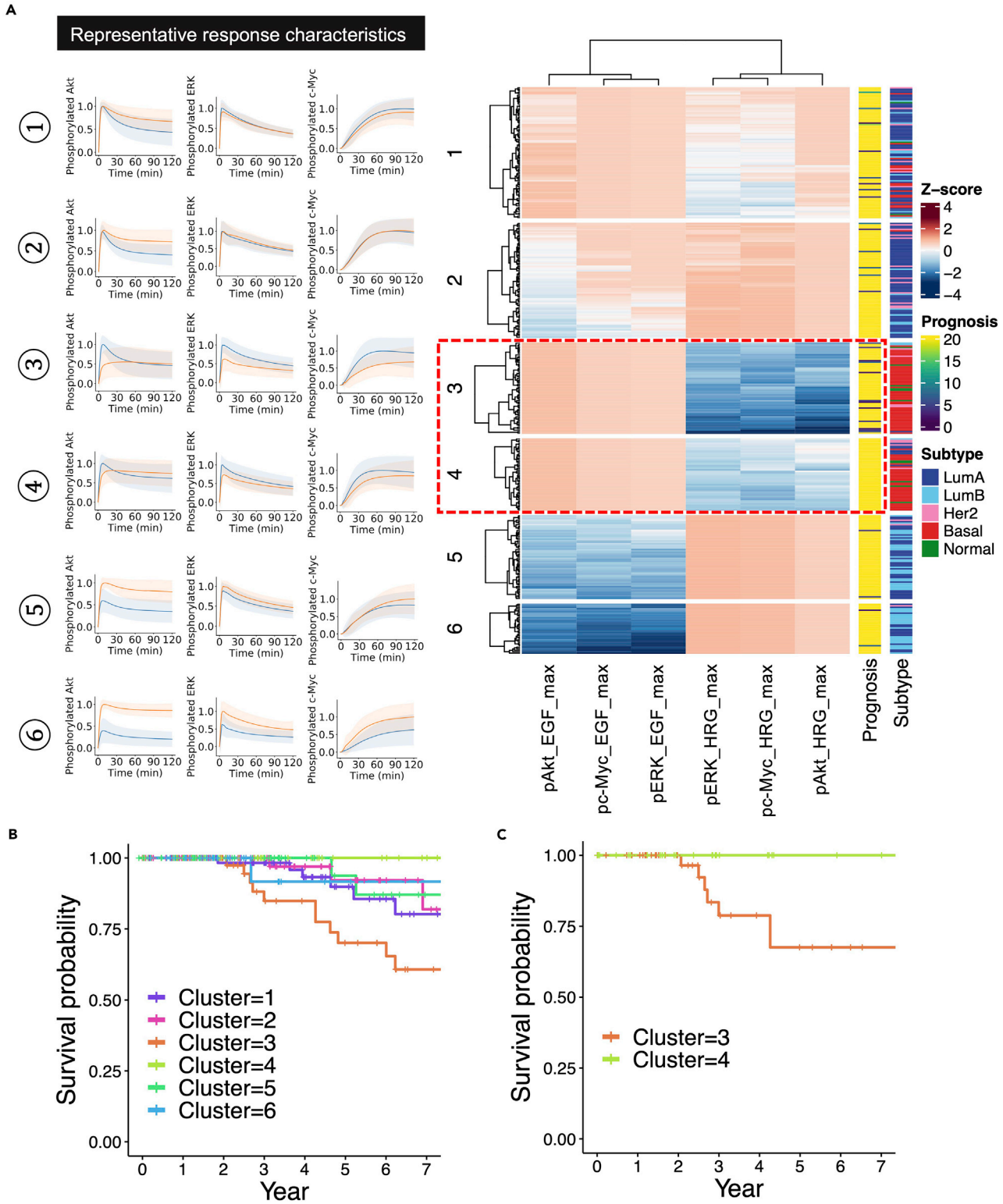


Figure 4. Stratification of breast cancer patients based on the predicted signaling dynamics

(A) The 369 breast cancer patients are classified based on personalized simulations. The prognostic score for patients who deceased within *n-1* to *n* years are denoted by n, and patients who were alive after 20 years are denoted in yellow. The representative signal response characteristics were

Figure 4. *Continued*
extracted from the topmost portion of each cluster. The blue and orange solid lines denote simulations with EGF and HRG stimulation, respectively. Shaded areas denote the SD. The box enclosed with the red dashed line indicates the two clusters in which basal-like patients are enriched.
(B and C) Kaplan-Meier survival curves of all patients for all clusters (B) and of patients with the TNBC subtype for clusters 3 and 4 (C).

17. Visualize the classification result as a heatmap.

```
>cd classification

# Rscript brca_heatmap.R [n_cluster: int] [figsize: tuple]

>Rscript brca_heatmap.R 6 8,5
```

EXPECTED OUTCOMES

Our results indicate that the ErbB signaling pathway model trained with phospho-protein time series data obtained from four breast cancer cell lines (Figure 3) can classify TNBC patients into two subgroups, namely patients with better prognosis and poor prognosis, when using maximum activation levels as dynamic characteristics for the stratification (Figure 4).

LIMITATIONS

In this protocol, mechanistic models were personalized using gene expression levels of each patient's transcriptomic data. However, gene mutations are closely linked to treatment strategies in some types of cancer, such as lung cancer (Collisson et al., 2014). Future studies are needed to adapt the model to a wider range of cancer types by incorporating mutational information.

Although we demonstrated how users can classify breast cancer patients making use of this framework, it can be applied to other types of cancer as well. Users may do so by modifying several parts of the code explained in the "transcriptomic data integration" section. For example, "BRCA" and "BREAST" for the arguments of outputSubtype() and downloadCCLE() functions, respectively, can be replaced with "LUAD" and "SCLC" for lung adenocarcinoma and small cell lung cancer. Training datasets and the model to use should be suited for the type of cancer of interest as well.

TROUBLESHOOTING

Problem 1

Cannot run `downloadTCGA` function (step 5).

Potential solution

The error is caused by the recent changes made in the GDC Data Portal, which affect TCGAbiolinks (Colaprico et al., 2016) as well. Use the code hosted in breast_cancer that has resolved this issue (Imoto and Yamashiro, 2022).

Problem 2

Run out of memory while executing "mergeTCGAandCCLE".

Potential solution

Up to this step, approximately 11 GB of memory space is required; if you are using R studio, R studio may have a memory limit. Please check the amount of memory available in your R environment.

Problem 3

Get errors related to TCGAbiolinks.

Potential solution

Due to the recent changes made in the GDC Data Portal, TCGAbiolinks v2.24 or higher is required. The version of TCGAbiolinks registered in Bioconductor may be out of date. Please check the version of TCGAbiolinks installed in your environment.

Problem 4

It takes too long to simulate patient-specific models (step 15).

Potential solution

You can specify the number of CPU cores in run() function for parallel execution of simulations.

Problem 5

Any other problems or errors are encountered.

Potential solution

Please head over to GitHub Issues (https://github.com/pasmopy/breast_cancer/issues) if you have discovered an error or need help.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources or reagents should be directed to and will be fulfilled by the lead contact, Mariko Okada (mokada@protein.osaka-u.ac.jp).

Material availability

No new unique reagents were generated as part of this study.

Data and code availability

The analysis code utilized in this study can be found at https://doi.org/10.5281/ZENODO.6781265. A previously published article (Imoto et al., 2022) includes all the datasets analyzed in this study.

ACKNOWLEDGMENTS

We thank Mr. Kiwamu Arakane for critically reading the manuscript. M.O. was supported by JSPS KAKENHI Grant Nos. 17H06299, 17H06302, and 18H04031; JST-CREST Grant No. JPMJCR21N3; JST-Mirai Program No. JPMJMI19G7; Moonshot R&D Grant No. JPMJMS2021; and the Uehara Memorial Foundation. H.I. was supported by a Grant-in-Aid for JSPS Fellows (Grant No. 20J20192).

AUTHOR CONTRIBUTIONS

H.I. and S.Y. developed the analytics pipeline. K.M. updated the code for transcriptomic data integration to be compatible with the newest GDC. M.O. supervised this study. H.I., K.M., and M.O. wrote the manuscript.

DECLARATION OF INTERESTS

A Japanese patent application (No. 2021-128753) related to this work was filed (H.I., S.Y., and M.O.).

REFERENCES

Barretina, J., Caponigro, G., Stransky, N., Venkatesan, K., Margolin, A.A., Kim, S., Wilson, C.J., Lehár, J., Kryukov, G.V., Sonkin, D., et al. (2012). The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity. Nature *483*, 603–607. https://doi.org/10.1038/nature11003.

Birtwistle, M.R., Hatakeyama, M., Yumoto, N., Ogunnaike, B.A., Hoek, J.B., and Kholodenko, B.N. (2007). Ligand-dependent responses of the ErbB signaling network: experimental and modeling analyses. Mol. Syst. Biol. *3*, 144. https://doi.org/10.1038/msb4100188.

Colaprico, A., Silva, T.C., Olsen, C., Garofano, L., Cava, C., Garolini, D., Sabedot, T.S., Malta, T.M., Pagnotta, S.M., Castiglioni, I., et al. (2016). TCGAbiolinks: an R/Bioconductor package for integrative analysis of TCGA data. Nucleic Acids Res. *44*, e71. https://doi.org/10.1093/nar/gkv1507.

Collisson, E.A., Campbell, J.D., Brooks, A.N., Berger, A.H., Lee, W., Chmielecki, J., Beer, D.G., Cope, L., Creighton, C.J., Danilova, L., et al. (2014). Comprehensive molecular profiling of lung adenocarcinoma: the cancer genome atlas research network. Nature *511*, 543–550. https://doi.org/10.1038/nature13385.

Durinck, S., Spellman, P.T., Birney, E., and Huber, W. (2009). Mapping identifiers for the integration of genomic datasets with the R/ Bioconductor package biomaRt. Nat. Protoc. *4*, 1184–1191. https://doi.org/10.1038/nprot.2009.97.

Gu, Z., Eils, R., and Schlesner, M. (2016). Complex heatmaps reveal patterns and correlations in multidimensional genomic data. Bioinformatics *32*, 2847–2849. https://doi.org/10.1093/bioinformatics/btw313.

Gu, Z., Gu, L., Eils, R., Schlesner, M., and Brors, B. (2014). Circlize implements and enhances circular visualization in R. Bioinformatics *30*, 2811–2812. https://doi.org/10.1093/bioinformatics/btu393.

Imoto, H., and Yamashiro, S. (2022). pasmopy/breast_cancer: protocol for stratification of TNBC patients. Zenodo. https://doi.org/10.5281/ZENODO.6781265.

Imoto, H., Yamashiro, S., and Okada, M. (2022). A text-based computational framework for patient -specific modeling for classification of cancers. iScience *25*, 103944. https://doi.org/10.1016/j.isci.2022.103944.

Imoto, H., Zhang, S., and Okada, M. (2020). A computational framework for prediction and analysis of cancer signaling dynamics from RNA sequencing data—application to the ErbB receptor signaling pathway. Cancers *12*, 2878. https://doi.org/10.3390/cancers12102878.

Nakakuki, T., Birtwistle, M.R., Saeki, Y., Yumoto, N., Ide, K., Nagashima, T., Brusch, L., Ogunnaike, B.A., Okada-Hatakeyama, M., and Kholodenko, B.N. (2010). Ligand-specific c-Fos expression emerges from the spatiotemporal control of ErbB network dynamics. Cell *141*, 884–896. https://doi.org/10.1016/j.cell.2010.03.054.

Purvis, J.E., and Lahav, G. (2013). Encoding and decoding cellular information through signaling dynamics. Cell *152*, 945–956. https://doi.org/10.1016/j.cell.2013.02.005.

Robinson, M.D., McCarthy, D.J., and Smyth, G.K. (2009). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics *26*, 139–140. https://doi.org/10.1093/bioinformatics/btp616.

Van Der Walt, S., Colbert, S.C., and Varoquaux, G. (2011). The NumPy array: a structure for efficient numerical computation. Comput. Sci. Eng. *13*, 22–30. https://doi.org/10.1109/MCSE.2011.37.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat. Methods *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2.

Waskom, M. (2021). seaborn: statistical data visualization. J. Open Source Softw. *6*, 3021. https://doi.org/10.21105/joss.03021.

Weinstein, J.N., Collisson, E.A., Shaw, K.R.M., Ellrott, K., Sander, C., Ellrott, K., Sander, C., Stuart, J.M., Chang, K., Creighton, C.J., et al. (2013). The cancer genome atlas pan-cancer analysis project. Nat. Genet. *45*, 1113–1120. https://doi.org/10.1038/ng.2764.

Zhang, Y., Parmigiani, G., and Johnson, W.E. (2020). ComBat-seq: batch effect adjustment for RNA-seq count data. NAR Genom. Bioinform. *2*, lqaa078. https://doi.org/10.1093/nargab/lqaa078.