GEARS - a toolbox for Global parameter Estimation with Automated Regularisation via Sampling

Jake Alan Pitt and Julio R. Banga
(Bio)Process Engineering group - IIM-CSIC,
Spanish National Research Council
c/Eduardo Cabello, 6. 36208, Vigo (Spain)
Email: jp00191.su@gmail.com, julio@iim.csic.es

CSIC
Consejo Superior de Investigaciones Científicas

November 20, 2018

Contents

1 Introduction

GEARS is a Matlab toolbox for parameter estimation in nonlinear dynamic models composed of deterministic ordinary differential equations (ODEs). In this estimation problem, we seek to minimise the distance between model predictions and observations (measurements). In other words, we want to perform model calibration with respect to specific data set(s).

GEARS is based on the combination of three main strategies: (i) global optimisation (to avoid convergence to local solutions), (ii) reduction of the search space (i.e. tighten bounds on parameters), (iii) regularised estimation, a strategy used to handle overfitting (i.e. fitting the noise rather than the signal). As a result, GEARS can avoid under and overfitting problems, and requires minimum supervision from the user. These capabilities are especially useful when calibrating ODE-based models with highly nonlinear and flexible dynamics (e.g. models of biological oscillators).

Considering a frequentist framework, the estimation problem can be formulated as an optimisation using a weighted nonlinear least-squares criterion (Eqn (1)), as derived from a maximum-likelihood formulation under the condition of Gaussian measurement noise. Mathematically the statement is:

$$\min_{\boldsymbol{\theta}} Q_{\mathrm{NLS}}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \sum_{k=1}^{n_e} \sum_{j=1}^{n_y} \sum_{i=1}^{n_t} \left(\frac{y_{k,j}(t_i, \boldsymbol{\theta}) - \widetilde{y}_{k,j,i}}{\sigma_{k,j,i}} \right)^2 \tag{1}$$

Subject to:

$$\frac{d\mathbf{x}(t,\boldsymbol{\theta})}{dt} = \mathbf{f}\left(t, \boldsymbol{\theta}, \mathbf{x}\left(t, \boldsymbol{\theta}\right)\right) \tag{2}$$

$$\mathbf{y}_k\left(t, \boldsymbol{\theta}\right) = \mathbf{g}_k\left(t, \boldsymbol{\theta}, \mathbf{x}\left(t, \boldsymbol{\theta}\right)\right) \tag{3}$$

$$\mathbf{x}\left(t_0, \boldsymbol{\theta}\right) = \mathbf{x}_{k,0} \tag{4}$$

$$\theta_i \in [\theta_i^{\min}, \theta_i^{\max}]\ \forall\, \theta_i \in \boldsymbol{\theta} \tag{5}$$

where $\mathbf{y}_k$ is the observation function of the kth experiment, $y_{k,j}(t_i, \boldsymbol{\theta})$ the predicted value at the ith time point for the jth observable and the kth experiment, $\widetilde{y}_{k,j,i}$ the experimentally measured data at the same time point, $\sigma_{k,j,i}$ the corresponding standard deviation, $\boldsymbol{\theta}^{\min}$ and $\boldsymbol{\theta}^{\max}$ the lower and upper bounds for the parameters, and $\mathbf{x}_{k,0}$ the initial conditions of the states for the kth experiment. If no $\boldsymbol{\sigma}$ is provided by the user GEARS scales the residuals to ensure that each observable is given equal weight in the estimation.

This class of problems can be extremely challenging due to several common pitfalls:

1. possible convergence to local optima (due to multimodality of the cost function), resulting in suboptimal solutions (underfitting) that might lead to the false conclusion that the model is incorrect

2. a lack of prior knowledge for (some or all) the parameters, leading to a huge range between the lower and upper (i.e. massive search spaces)

3. possible overfitting, i.e. when the fit to noisy data is extremely good but the estimation has fitted the noise rather than the signal. As a consequence, the calibrated model will exhibit poor generalisation capabilities (low predictive power)

To overcome these issues, GEARS is based on a two-phase optimisation workflow combining several computational strategies:

1. efficient global optimisation, using a hybrid metaheuristic (enhanced scatter search) that makes use of a combination of robust global search and an efficient gradient-based local search method (based on adaptive nonlinear least squares).
2. the reduction of the search space via an initial sampling by scatter search, followed by parameter bounding based on thresholding the parameter-cost distribution.
3. self-tuning regularisation, where the initial sampling by scatter search is used to automatically tune the regularisation in the second optimisation phase without introducing an additional computational cost.

The first optimisation phase performs a standard global optimisation using the cost function defined in Eqn (1). The second optimisation phase (regularised estimation step) uses a Tikhonov term in an extended cost function:

$$\widehat{Q}_{\text{NLS}}(\boldsymbol{\theta}) = Q_{\text{NLS}}(\boldsymbol{\theta}) + \alpha(\boldsymbol{\theta} - \boldsymbol{\theta}^{ref})^T \mathbf{W}^T \mathbf{W} \left(\boldsymbol{\theta} - \boldsymbol{\theta}^{ref}\right) \tag{6}$$

where α is a strictly positive regularisation parameter, $\boldsymbol{\theta}^{ref}$ is a reference parameter vector, and $\mathbf{W}$ is a diagonal scaling matrix $W_{i,i} = \frac{1}{\theta_i^{\text{ref}}}$ and $W_{j\neq i,i} = 0$. GEARS will tune the α and $\boldsymbol{\theta}^{ref}$ automatically, although users can provide the latter if desired (e.g. in situations where good prior knowledge regarding the parameters is available).

An overview of the workflow used in GEARS is shown in Figure 1.

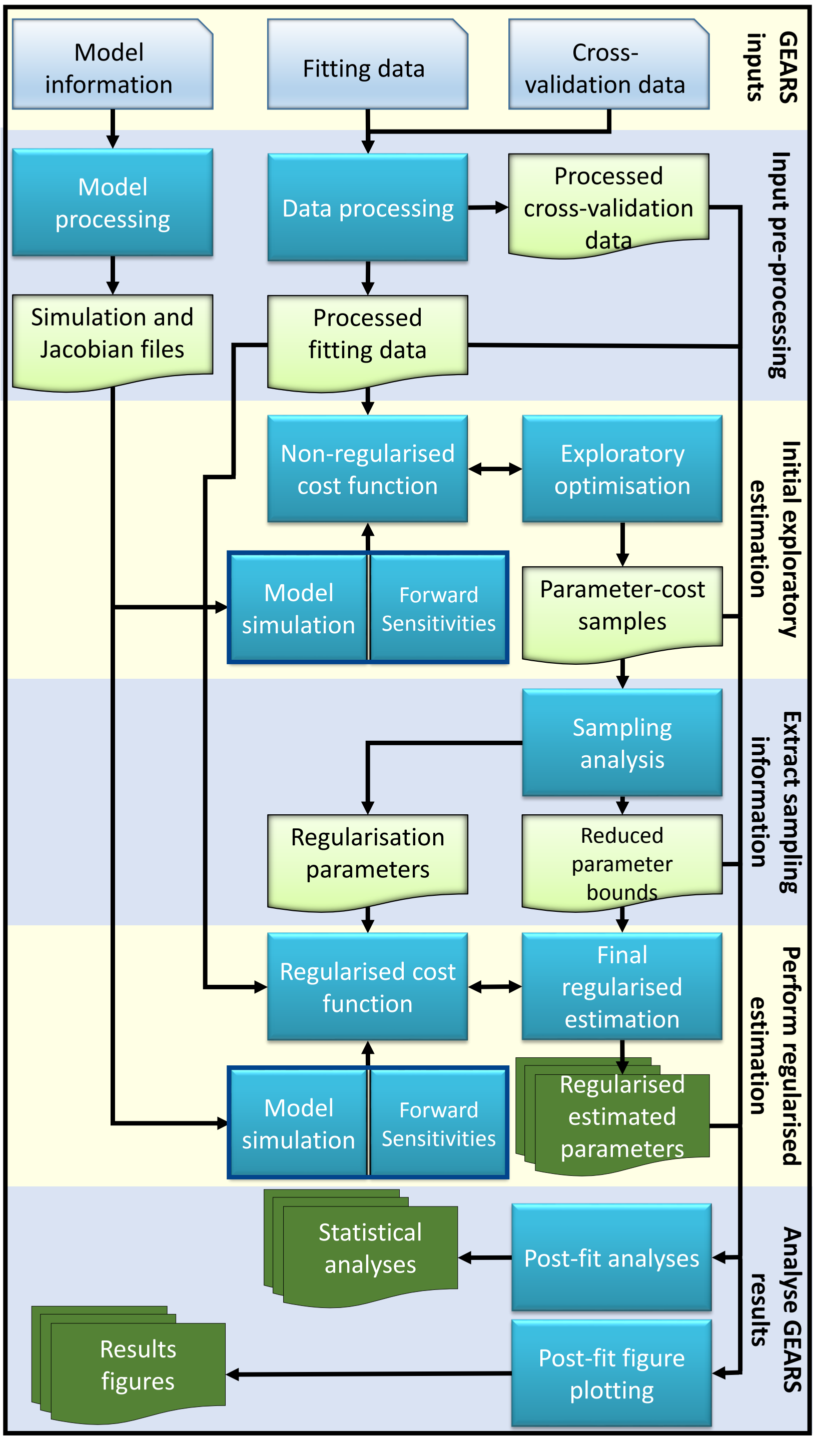


Figure 1: An overview of the GEARS procedure, where light blue nodes denote inputs, dark blue nodes processes, light green intermediate results and dark green final results.

2 Licensing

GEARS is distributed under the GNU General Public License version 3 (GPL v3) `http://www.gnu.org/licenses/gpl.html`. Copyright 2018 Jake Alan Pitt and Julio R. Banga

3 Citing

If you use this toolbox and publish the results, please cite it with the following references.

Regarding the methodology:
Pitt, J.A. and Banga, J.R. (2018) Parameter estimation in models of biological oscillators: an automated regularised estimation approach. **Submitted**.

Regarding the software:
Pitt, J.A. and Banga, J.R. (2018) GEARS - a toolbox for Global parameter Estimation with Automated Regularisation via Sampling.
DOI 10.5281/zenodo.1420464

4 Download and installation

The latest version of `GEARS` can be downloaded from:

`https://japitt.github.io/GEARS/`

`GEARS` runs on Matlab R2015b or later and is multi-platform (Windows and Linux). Both the optimisation and symbolic mathematics Matlab toolboxes are required to run `GEARS`. In particular, `GEARS` has been tested under the following operating system and Matlab version combinations:

- Windows 7 with Matlab 2015b, Matlab 2016b and Matlab 2017b
- Windows 10 with Matlab 2016a
- Linux (distribution Rocks Clusters 6.1.1, Sand Boa, based upon CentOS 6.5) with Matlab 2016a

The `GEARS` distribution includes an special version of the MEIGO toolbox (Egea et al., 2014). In addition to this, `GEARS` also requires that the following software is installed prior to its use:

1. `AMICI` toolbox, available at `http://icb-dcm.github.io/AMICI/`. *AMICI* is used for efficient computation of parametric sensitivities and model simulation(Fröhlich et al., 2016).
2. `Ghostscript`, available at `https://www.ghostscript.com`, is required to create PDF reports of results.

Once these dependencies are installed and tested, GEARS can be installed by simply unzipping the package into a directory and adding its folders (and subfolders) into the Matlab path by running the script:

```
>> Add_GEARS_paths.m
```

As a final step, GEARS should be tested by running the included examples.

5 Using GEARS

In order to solve a parameter estimation problem using GEARS, users must write input files describing (i) the dynamic model, the parameters to be estimated and their bounds; (ii) the data for fitting and (optionally) cross-validation. Additionally, users might want to change the default values for the different solver options (e.g. stopping criteria, local solvers, etc.). Finally, users can also change the default options regarding the reporting and plotting of the estimation results.

5.1 Defining the problem inputs (model and data)

The first step in using GEARS is to the define the problem inputs, which are used to define the dynamic model (as a set of ODEs) and the data to be used in the parameter estimation:

- Model_information is a Matlab structure containing the model specific information for the problem with the following fields:

```
.Model_name % The name of the model, which will be used for ...
    identification. (string)
.Diff_eqns  % The model equations. Should be of the format {'dx = y...
    ^2'; 'dy = x'}. Equations not beginning with d will be ...
    considered non-differential. (cell array)
.Var_names % The names of the states in the differential equations....
     (cell array)
.Fitting_params % The names of the parameters to to be estimated. (...
    cell array)
.Fixed params % The names of fixed parameters. The value should be ...
    defined in data structures in the same order. (cell array) ...
    [Optional]
.Param_bounds_lower % The initial lower bounds for the fitting ...
    parameters. It should be in the same order as Fitting_params. (...
    vector)
.Param_bounds_upper % The initial upper bounds for the fitting ...
    parameters. It should be in the same order as Fitting_params. (...
    vector)
```

- Fitting_data is a structure containing the data that will be used for the fitting, organised by experiment. It contains the following fields:

```
."experiment name" % The experiment name. (string) E.g. ...
    Fitting_data.exp1
."experiment name".Time_points % The time point of each measurement...
     (vector)
."experiment name".Measurements % A matrix with the measurements, ...
    with dimensions # Time points by # Observables. Missing points ...
    can be denoted as NaN. (matrix)
."experiment name".Standard_deviation % A matrix with the standard ...
    deviations for each data point. Same size as .Measurements. (...
    matrix)
."experiment name".Observables % The names of the observed states, ...
    assuming a direct mapping. If the observation function is more ...
    complicated, use .Custom_function instead. (cell array)
."experiment name".Custom_function % For observation functions that...
     are not direct mappings of states {'x1^2 - 1'; 'x2 - 69'} for ...
    example. (cell array) [Optional]
."experiment name".Initial_conditions % The initial conditions of ...
    the experiment for each state, in the same order as Var_names. ...
    (vector)
."experiment name".Fixed_params % The values of fixed parameters (...
    which will not be estimated), in the same order as any ...
    Fixed_params (vector) [Optional]
```

- `Validation_data` is a structure with data that will be used for cross-validation purposes (i.e. evaluating the fit of the calibrated model with a second set of data not used during the fitting). The same format as `Fitting_data` should be used. This input is optional but if this structure is not declared no model cross-validation will be performed.

- `Results_folder` is a string indicating the directory where results will be stored

5.2 Setting solvers and output options

Default options are given in `GEARS_Options.m`. Users can modify the options for each run by simply editing the corresponding values. We describe the available options below. To create copies of the options file and then run `GEARS` using the options from the copy file `GEARS` can be set to use alternative options scripts (that contain the same options) by initiating `GEARS` with the additional input Alternative_options_script as follows:

```
>> Results = Run_GEARS(Results_folder, Model_information, Fitting_data...
, Validation_data, 'Alternative_options_script')
```

Where Alternative_options_script will be the name of the copy of GEARS_options you wish to use.

5.2.1 Numerical solvers options

Broadly speaking, `GEARS` solves the parameter estimation problem using a single shooting approach, i.e. an optimiser iteratively generates points in parameter space, which are evaluated by solving the corresponding initial value (IVP) problem. GEARS uses the eSS method in MEIGO as the global optimiser, and the AMICI toolbox to solve the IVP. The default solvers options are:

- AMICI options

```
Int_opts.atol = 1E-8; % Absolute integration tolerance. (scaler)
Int_opts.rtol = 1E-8; % Relative integration tolerance. (scaler)
Int_opts.maxsteps = 1E5; % Maximum number of integration steps. (...
    scaler)
```

- MEIGO options, where stopping criteria are set by values to reach (vtr), maximum number of evaluations (maxeval) and maximum CPU time (maxtime). Stopping criteria in parameter estimation is still an open question but insights on such values can be found by performing initial runs of the problem.

```
MEIGO_options.maxeval = []; % Maximum number of function ...
    evaluations. (scaler)
problem.vtr = []; % The cost value of the optima. If the cost is ...
    known this can be used. (scaler) [Optional]
MEIGO_options.maxtime = []; % Maximum CPU time in seconds. If not ...
    finite custom convergence criteria will be used. (scaler) ...
    [Optional]
MEIGO_options.log_var = 1:length(Model_information.Fitting_params);...
     % The indices of variables that should be searched for in log ...
    space.
MEIGO_options.log_level = 2; % What should be considered in log ...
    space: 1 - Only ref set, 2 - Everything expect the local search...
    , 3 - Everything.
MEIGO_options.local.solver = 'nl2sol'; % Local solver. (string)
MEIGO_options.local.n1 = 1; % Number of iterations before the first...
     local search. (scaler)
MEIGO_options.local.n2 = 10; % Number of iterations between the two...
     local searches. (scaler)
MEIGO_options.local.finish = MEIGO_options.local.solver; % Local ...
    solver for the final search. (string)
```

5.2.2 Output and results options

The default plotting, post analysis and report options are:

- Plotting options

```
Plot_trajectories = true; % If the trajectories should be plotted. ...
    (logical)
Plot_samples = true; % If the parameter samples should be plotted. ...
    (logical)
Plot_residuals = true; % If the normalised residuals should be ...
    plotted. (logical)
Plot_pred_vs_meas = true; % If the predictions should be plotted ...
    against the measurements. (logical)
Plot_correlation_matrix = true; % If the correlation matrix should ...
    be calculated. (logical)
Plot_bounds_param_confidence = true; % If the parameter confidence ...
    intervals and parameter bounds should be plotted, turn on/off ...
    here. Requires Calculate_parameter_confidence to be true. (...
    logical)
Plot_trajectories_with_uncertainty = true; % % If the trajectories ...
    should be plotted with uncertainty intervals. (logical)
Plot_convergence_curves = true; % If convergence curves should be ...
    plotted. (logical)
Plot_regularised_parameter_summary = true; % If the summary of the ...
    estimation should be created in a uitable, recommended if using...
     Create_figure_report. (logical)
Max_number_open_figures = 0; % The maximum number of figures that ...
    will be open at one time (to prevent huge numbers of figures ...
    being opened). All figures are still saved. (scaler)
Max_number_subplots = 8; % The maximum number of subplots that ...
    should be contained in a single figure. (scaler)
```

- Post analysis options

```
Calculate_NRMSE = true; % If the normalised root mean square error ...
    should be calculated. (logical)
Calculate_R2 = true; % If R2 goodness of fit metrics should be ...
    calculated. (logical)
Calculate_chi2 = true; % If the chi2 hypothesis should be tested. ...
    Requires the statistics toolbox (logical)
Calculate_parameter_confidence = true; % If the confidence for the ...
    parameters should be calculated using the FIM (logical)
```

- Report options

```
GEARS_diary = true; % If GEARS should save a diary of the ...
    procedure. (logical)
Export_results_to_xls = true; % If GEARS results should be exported...
     to xls. (logical)
Export_results_to_html = true; % If GEARS results should be ...
    exported to html. (logical)
```

```
Create_figure_report = true; % If GEARS should create a pdf report ...
    containing all figures. Requires Ghostscript. (logical)
Ghostscript_path = 'C:\Program Files (x86)\gs\gs9.19\bin\...
    gswin32c.exe'; % The path to the Ghostscript exe in your ...
    machine. Only required for the figure report. Not required for ...
    Linux. (string) [Optional]
```

- Other options

```
P_ref = []; % If you already have a specific reference value you ...
    want to use. (vector) [Optional]
Seed_to_use = []; % If you want to start the optimisation at a ...
    specific seed. (seed) [Optional]
Format_to_use = 'shorte'; % The display format matlab should use ...
    for the GEARS procedure. (string) [Optional]
```

5.3 Executing GEARS

Once the above inputs are defined, the **GEARS** procedure is initiated by executing the following main function:

```
>> Results = Run_GEARS(Results_folder, Model_information, Fitting_data...
, Validation_data)
```

5.4 GEARS results

In addition to the parameter estimation, the toolbox also performs the following tasks:

- Post-fit analyses
  - Normalised root mean square error (NRMSE)
  - R^2 test
  - χ^2 test
  - Parameter uncertainty (using Fisher information)
  - Correlation matrix (using Fisher information)
  - Active bounds
- Post-fit plotting
  - Trajectories for fitting and cross-validation
  - Parameter space samples
  - Visualisation of parameter bound reduction
  - Residuals
  - Predictions vs measurements

  - Trajectory uncertainty (via Fisher information)
  - Convergence curves

- Results reports
  - HTML markup of results
  - xls markup of results
  - Combined pdf report of all plotted figures

Important: caveats about the FIM condition number

In GEARS we use the Fisher information matrix (FIM) to calculate metrics such as the confidence intervals or the correlation matrix. Users need to take into account two important issues:

1. the FIM is a first order approximation, so in the presence of highly nonlinear terms, inaccuracies may occur
2. some of the metrics calculated rely on calculating the inverse of the FIM, and this can be problematic when the FIM is badliy conditioned or singular (which is typical in the presence of lack of identifiability)

GEARS reports the condition number of the FIM (in the figure report and the Matlab screen output). Users need to be especially careful when the FIM is singular or near singular (the larger the condition number the closer the FIM is to being singular). When the condition number is higher than 10^5, the metrics dependent on the FIM are most likely numerical artefacts and no longer hold much meaning. These situations are likely caused by a lack of identifiability.

5.4.1 The results structure

All results from GEARS are stored in the results structure which has the following fields:

- Results

```
.Total_time_taken % The time taken to run the whole GEARS ...
    procedure.
.Markup_tag % The tag used in the markup reports for ...
    identification.
.Simulation_handle % An anonymous functions that can be used to ...
    simulate the system (see 6.1)
.Results_folder % A hyperlink to the results folder.
```

- Results.Model_information and Results.Settings_used both simply store information set by the user for the GEARS analysis.

- Results.Parameter_bounding stores the original bounds provided by the user and the reduced bounds calculated in `GEARS`.
- Results.Regularisation stores the regularisation parameters tuned by `GEARS`.
- Results.Sampling

```
.Parameter_sample_points % The time parameter points that were ...
    sampled in the exploratory estimation. (matrix)
.Parameter_sample_costs % The costs of the parameter points sampled...
     in the exploratory estimation. (vector)
.Sample_size % The number of points sampled. (scaler)
.Cost_cut_offs % The cost−cut−offs for each parameter. (vector)
```

- Results.Global_parameter_estimation

```
Results.Global_parameter_estimation.Seed % Stores the Seed used in ...
    the optimisation procedure. (seed)
Results.Global_parameter_estimation.Non_regularised_estimation % ...
    Stores the MEIGO results from the exploratory estimation.
Results.Global_parameter_estimation.Regularised_estimation % Stores...
     the MEIGO results from the regularised estimation.
```

- Results.Statistics stores the results of the statistical tests requested in the options i.e. χ^2, R^2 tests.
- Results.Warning_log stores all the warnings that were found in the procedure.

5.4.2 Figures plotted

`GEARS` saves all figures created in a subfolder ”Figures” of the requested results folder. The figures plotted are as follows:

- Bounds_param_confidence_plot: A plot showing both the reduction in parameter bounds as well as the parameter confidence intervals calculated.
- Convergence_curves: A plot of the convergence curves of the regularised estimation for both CPU time and function evaluations.
- Fitting_figures: A plot of the regularised and non-regularised fits to the fitting data for each experiment.
- Validation_figures: A plot of the regularised and non-regularised predictions to the each cross-validation data experiment.
- Regularised_fit_with_uncertainty and Non_regularised_fit_with_uncertainty: Plots of the regularised and non-regularised fits with uncertainty intervals respectively.

- Regularised_validation_with_uncertainty and Non_regularised_validation_with_uncertainty: The predictions given by the regularised and non-regularised estimations with uncertainty intervals respectively.

- Regularised_fitting_corr_matrix and Non_regularised_fitting_corr_matrix: The correlation matrices of the regularised and non-regularised fits plotted as heat-maps respectively.

- Regularised_fitting_pred_vs_meas and Non_regularised_fitting_pred_vs_meas: The normalised predicted values in the regularised and non-regularised fits plotted against the fitting data respectively.

- Regularised_validation_pred_vs_meas and Non_regularised_validation_pred_vs_meas: The normalised predicted values in the regularised and non-regularised predictions plotted against the cross-validation data respectively.

- Regularised_fitting_residuals and Non_regularised_fitting_residuals: The normalised residuals of the regularised and non-regularised fits respectively.

- Regularised_validation_residuals and Non_regularised_validation_residuals: The normalised residuals of the regularised and non-regularised predictions respectively.

- Sample_plot: A plot of the parameter-cost distribution of each parameter with its' reduced bounds and cost cut off box.

- Regularised_parameter_summary: A uitable containing the summary of the final regularised estimation results.

5.5 Auxiliary functions

5.5.1 Simulating the system from the results structure

The model can be simulated directly from the GEARS results structures using the Results.Simulation_handle handle:

```
>> Results.Simulation_handle(Param_values, Processed_fitting_data."experiment_name...
", Results.Settings_used.AMICI_options, Time_points)
```

This will simulate the model for the initial conditions in Processed_fitting_data."experiment_name".

5.5.2 Running a single global optimisation

The full GEARS procedure uses two sequential global optimisation steps. If desired, users can run a single non-regularised global optimisation using the eSS solver by using the following option in the options script:

```
Run_standard_optimisation = true;
```

5.5.3 Running a multi-start optimisation

Sometimes users might want to compare the results obtained using global optimisation with those from a multi-start of local methods. In GEARS this can be done by using the following option in the options script:

```
Run_multi_start = true;
```

5.5.4 Creating GEARS reports outside the procedure

Users may wish to create the GEARS outside of a run of the entire run of the GEARS procedure. This can easily be done by running each reports respective function.

- For the GEARS xls summary use the following code:

  ```
  >> Write_GEARS_results_xls_summary(Results, Processed_fitting_data...
  , 'xls_summary', Processed_validation_data)
  ```

  where Results, Processed_fitting_data and Processed_validation_data are directly output from the GEARS analysis and 'xls_summary' is the file name which the xls summary will be saved.

- For the GEARS html summary use the following code:

  ```
  >> Write_GEARS_results_html_summary(Results, Processed_fitting_data...
  , 'html_summary', Processed_validation_data), 'Figures')
  ```

  where Results, Processed_fitting_data and Processed_validation_data are directly output from the GEARS analysis and 'html_summary' is the file name which the html summary will be saved. The 'Figures' input is the folder name that contains the figures created by GEARS that should be included.

- For the GEARS figure report use the following code:

  ```
  >>Figure_report_GEARS(Figure_list, 'GEARS_figure_report', Ghostscript_path...
  , Results.Markup_tag)
  ```

  where Figure_list is the list of figures to be included in the report, 'GEARS_figure_report' is the file name the report will be saved as and Ghostscript_path is the path to the Ghostscript exe file (use [ ] for Linux). Here the Results.Markup_tag is the markup tag used for identification, if this is left empty no tags will be put on the figures. It is possible to simplify the process of creating the figure list using the following code:

  ```
  >> Figure_list = Find_files_in_folders_GEARS('Figures', '.fig')
  ```

  where 'Figures' is the folder containing the figures. Please note that this list will be ordered alphabetically.

6 Example: The Fitzhugh-Nagumo oscillator

One of the examples in `GEARS` is of the parameter estimation of the Fitzhugh-Nagumo model (FHN). The model is given by the following system:

$$\frac{dV}{dt} = g\left(V - \frac{V^3}{3} + R\right) \tag{7}$$

$$\frac{dR}{dt} = -\frac{1}{g}\left(V - a + b \cdot R\right) \tag{8}$$

$$V(t_0, \boldsymbol{\theta}) = V_0,\ R(t_0, \boldsymbol{\theta}) = R_0 \tag{9}$$

$$y(t_i) = V(t_i) \tag{10}$$

$$\boldsymbol{\theta} = \{a, b, g\} \in [10^{-5}, 10^{5}] \tag{11}$$

Synthetic data was generated for the Fitzhugh-Nagumo model for parameter values $\{a, b, g\} = \{0.2, 0.2, 3\}$ for the initial conditions $V_0 = -1, R_0 = 1$. This data was generated with a standard deviation of 10% of the nominal signal level and a detection threshold of 0.1. This set-up for generating data was used to set up one fitting set of data and two data sets for cross-validation. Initial conditions for the cross-validation sets were varied randomly within a meaningful range.

6.1 Coding the model structure

The Fitzhugh-Nagumo model structure and parameter bounds are implemented in `GEARS` as follows bellow:

- The Model_information structure contains all of the information specific to the model. Here we don't have any fixed parameters so that field is not required.

```
Model_information.Model_name = 'FHN';
Model_information.Diff_eqns = {'dV = g*(V - V^3/3 + R)'; 'dR = (-1/...
    g)*(V - a + b*R)'};
Model_information.Var_names = {'V', 'R'};
Model_information.Fitting_params = {'a', 'b', 'g'};
Model_information.Param_bounds_lower = [10^-5 10^-5 10^-5];
Model_information.Param_bounds_upper = [10^5 10^5 10^5];
```

6.2 Encoding data into `GEARS`

Data in `GEARS` is written into two structures: Fitting_data and Validation_data. The Fitting_data contains all the data sets that will be fit to while Validation_data contains data that will only be used for cross-validation. It is important to note that the Validation_data will not be used during the optimisation stage of GEARS (the actual estimation of the parameters) and will only be considered once parameter values have been estimated. Both structures have the same format and for the FHN model are encoded as seen below.

- Fitting_data contains all of the data that we are going to fit to. Here we only have a singular experiment.

```
Fitting_exp_name = 'Fit_exp'; % The only fitting experiment
Fitting_data.(char(Fitting_exp_name)).Observables = {'V'};
Fitting_data.(char(Fitting_exp_name)).Time_points = [1.0000;  4...
    .8000; 8.6000; 12.4000; 16.2000; 20.0000];
Fitting_data.(char(Fitting_exp_name)).Measurements = [1.4480; 1...
    .0070; -1.3020; 1.6260; -1.3930; 1.8890];
Fitting_data.(char(Fitting_exp_name)).Standard_deviation = [0.2090;...
     0.1445; 0.1587; 0.1851; 0.2025; 0.2144];
Fitting_data.(char(Fitting_exp_name)).Initial_conditions = [-1 1];
```

- Validation_data contains the data for cross-validation. Here we have two different experiments which are each input under a different field of Validation_data with then all the information for each experiment being stored within that field.

```
Validation_exp_name = 'Val_exp_1'; % The 1st cross-validation ...
    experiment
Validation_data.(char(Validation_exp_name)).Observables = {'V'};
Validation_data.(char(Validation_exp_name)).Time_points = [1.0000; ...
    7.8000; 14.6000; 21.4000; 28.2000; 35.0000];
Validation_data.(char(Validation_exp_name)).Measurements = [-1.0670...
    ; -2.0690; 1.1710; 1.7200; -1.1620; -2.0670];
Validation_data.(char(Validation_exp_name)).Standard_deviation = [0...
    .1596; 0.2194; 0.1629; 0.2137; 0.1451; 0.2149];
Validation_data.(char(Validation_exp_name)).Initial_conditions = ...
    [-7.4850; 0.1792];
Validation_exp_name = 'Val_exp_2'; % The 2nd cross-validation ...
    experiment
Validation_data.(char(Validation_exp_name)).Observables = {'V'};
Validation_data.(char(Validation_exp_name)).Time_points = [1.0000; ...
    7.8000; 14.6000; 21.4000; 28.2000; 35.0000];
Validation_data.(char(Validation_exp_name)).Measurements = [1.9900;...
     -1.7880; 1.3180; 1.9030; -1.0800; -2.2870];
Validation_data.(char(Validation_exp_name)).Standard_deviation = [0...
    .2599; 0.2179; 0.1667; 0.2162; 0.1514; 0.2175];
Validation_data.(char(Validation_exp_name)).Initial_conditions = ...
    [-8.7420; 2.3690];
```

6.3 Setting the results save location

Results are saved in a folder that should be given by the user. If the folder is not found, it will be created by GEARS. Please note, if the results folder name is not changed in a new run, results from the previous run will be overwritten. Users are advised to change the name from run to run, either manually or automatically (e.g. via a unique ID label generated for each run). We define the results folder for the FHN example as indicated below.

- Results_folder gives the location that all results will be saved in.

```
Results_folder = 'FHN_results';
```

6.4 Running GEARS

Now that we have defined all of the GEARS inputs for the FHN example, the GEARS procedure can be initiated.

- Execute GEARS using the ”Run_GEARS” function.

```
Results = Run_GEARS(Results_folder, Model_information, Fitting_data...
    , Validation_data);
```

6.5 Results

Once the GEARS analysis has been performed we receive a number of different results. The most immediate of said results will be a summary of the final estimated parameters values and some key details (this will only be displayed in Matlab for models with ten or fewer parameters but always saved). Here we can see the regularised parameter values estimated within GEARS as well as some brief details as seen below.

Parameter	Value	Confidence (95%)	Coeff of variation (%)	Bounds status
a	1.5482e-01	± 8.8454e-03	2.9150e+00	Bounds not active
b	5.5983e-01	± 2.8386e-02	2.5870e+00	Bounds not active
g	2.7406e+00	± 9.0370e-02	1.6824e+00	Bounds not active

Table 1: A summary of the regularised results from the GEARS analysis of the FHN model

Important: always check the FIM condition number
Remember that in GEARS we use the Fisher information matrix (FIM) to calculate metrics such as the confidence intervals and the correlation matrix. These metrics can be wrong if the FIM is singular or near singular. Always check the condition number

reported to asses their reliability. As a rule of thumb, when the condition number is higher than 10^5, the metrics dependent on the FIM are most likely numerical artefacts. Typically this is caused by a lack of identifiability.

One key result is the plotting of the trajectories with confidence intervals, allowing the easy visualisation of the quality of fit/ prediction. GEARS will plot the trajectories for all fitting and cross-validation experiments dependant on the initial conditions of each experiment. The trajectories for the singular fitting experiment and the two cross-validation experiments for the FHN model (as defined in sections 6.1 and 6.2) with the estimated regularised parameter values as seen in table 1 can be seen in figure 2.

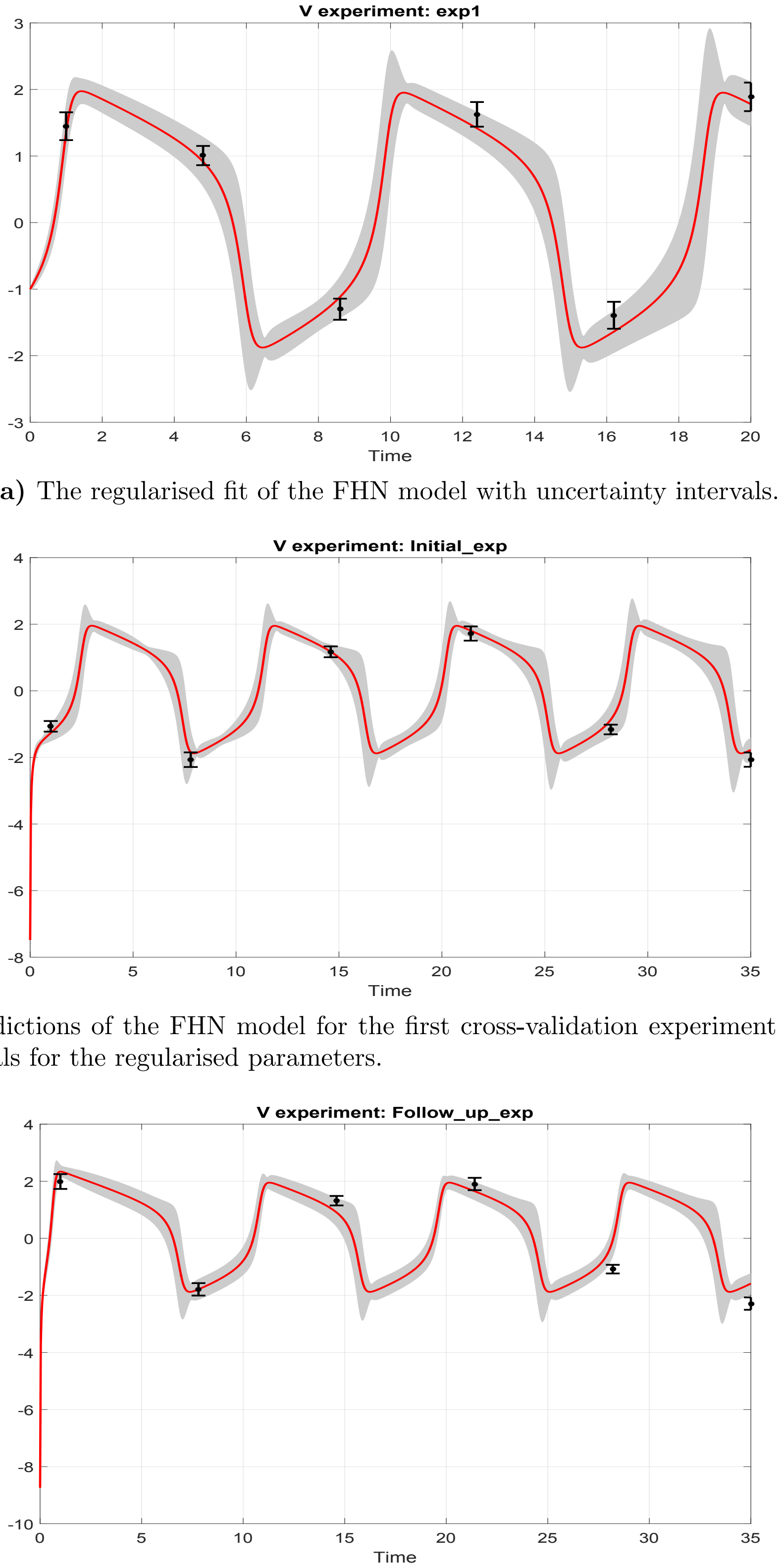
Another key result that can show the effectiveness of the regularisation procedure is the fitting figures containing both the regularised trajectories and the non-regularised trajectories. This allows the clear visualisation of the effect of the regularisation performed within GEARS and also allows the user to easily determine the level of overfitting that would have occurred without said regularisation. Here GEARS also plots the results for all experiments considered. The figures for the FHN example can be seen in figure 3. The improvement in the quality of cross-validation can also be seen in the statistics performed by GEARS such as the normalised root mean square error as seen in table 2 while the decrease in quality of fit can be seen in table 3.

NRMSEs	Regularised estimation	Non-regularised estimation
All experiments	5.8761e-01	1.5691e+00
Initial exp	3.1972e-01	1.8314e+00
Follow up exp	7.6703e-01	1.2531e+00

Table 2: The normalised root mean square error (NRMSE) for the cross-validation of the FHN model fit.

NRMSEs	Regularised estimation	Non-regularised estimation
exp1	2.7798e-01	1.0946e-01

Table 3: The normalised root mean square error (NRMSE) for the fitting of the FHN model.

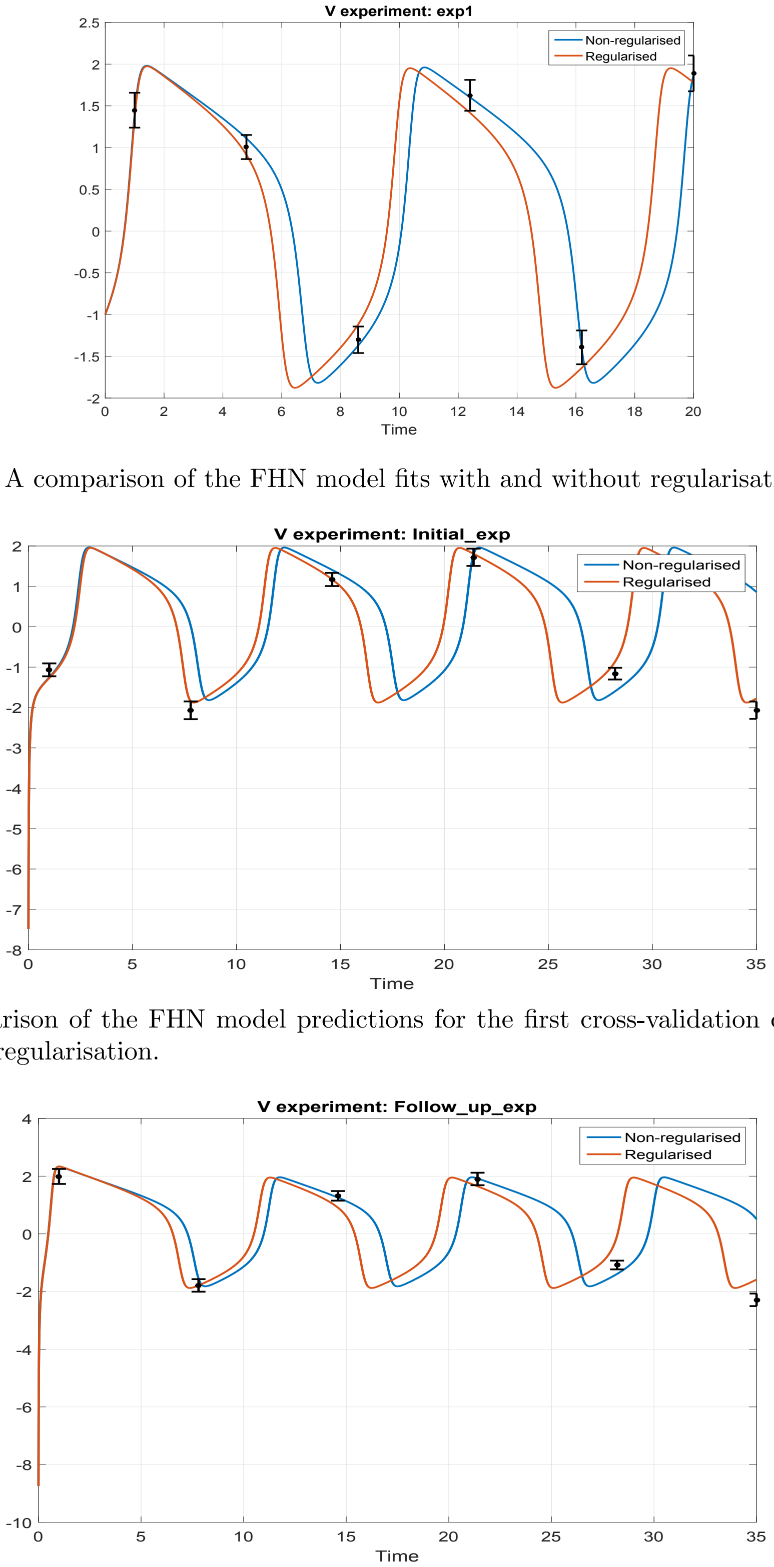


(a) The regularised fit of the FHN model with uncertainty intervals.

(b) The predictions of the FHN model for the first cross-validation experiment with uncertainty intervals for the regularised parameters.

(c) The predictions of the FHN model for the second cross-validation experiment with uncertainty intervals for the regularised parameters.

Figure 2: The trajectories for fitting and cross-validation of the FHN model with regularised parameter as calculated by `GEARS`



(a) A comparison of the FHN model fits with and without regularisation.

(b) A comparison of the FHN model predictions for the first cross-validation data set with and without regularisation.

(c) A comparison of the FHN model predictions for the second cross-validation data set with and without regularisation.

Figure 3: Figures showing the comparison between the regularised and non-regularised fits for both fitting and cross-validation.

7 Reproducibility

Internally GEARS uses two sequential steps of a stochastic global optimisation solver (enhanced scatter search as part of the MEIGO toolbox). In principle, the behaviour of stochastic solvers can be reproduced by fixing the pseudo-random number generator seed. Unfortunately in GEARS we cannot guarantee this kind of reproducibility. This is due to the fact that GEARS uses the initial optimisation stage as a sampling of parameter space. This sample can have some variations in size due to small variations in the CPU time taken (or the number of function evaluations) in each iteration. This, in turn, can be passed into the regularisation tuning and the final optimisation, so results will be slightly different. This potential lack of reproducibility is an inherent problem of most stochastic solvers.

8 Identfiability analysis with STRIKE-GOLDD and VisId

GEARS can export models to both STRIKE-GOLDD (Villaverde et al., 2016) and VisId (Gábor et al., 2017) formats, allowing for structural and practical identifiability analysis respectively. In order to use STRIKE-GOLDD for structural identifiability analysis (which should be done before any estimation), the model declared in GEARS can be exported as follows:

```
>> Convert_to_STRIKE_GOLDD_model_GEARS(Model_information, Fitting_data,...
 'STRIKE_GOLDD_model')
```

Where the file saved under the name STRIKE_GOLDD_model is directly usable in STRIKE-GOLDD.In order to use VisId for practical identifiability analysis, users need to perform an estimation with GEARS first and then export the model and results structure by doing:

```
>> Convert_to_VisId_model_GEARS(Results, Fitting_data, 'VisId_model')
```

Where the file saved as VisId_model is directly usable in VisId.

9 Known errors and solutions

There are some known minor errors (or warnings) within GEARS please consult this section for information regarding them and for information on how to solve said issues. Please be sure to consult this section before contacting the authors. The warnings output can be found in the warning log in either the results structure or the html summary.

9.1 Number of observables in the data matrix does not match the model

One known error is regarding the number of observables. This is an error that will only appear if AMICI has been used for the same model outside of GEARS. The error occurs due to the model name already being used to create AMICI files but with a different number of observables. The GEARS process clashes with the already existing files.

9.1.1 Error

This error will give out an error message in Matlab once the optimisation begins and the first simulation is attempted.

- The error message will appear as shown below.

```
Error using ami_RP_model
Number of observables in data matrix (0) does not match model
ny (1)

Error in simulate_RP_model (line 199)
sol = ami_RP_model(tout,theta(1:10),kappa(1:0),options_ami, ...
plist,pbar,xscale,data);
```

Where RP will be the model name defined in the model information structure.

9.1.2 Solution

This error has a simple solution, just follow the steps below.

- Delete the AMICI model folder for the model with the error in AMICI/models.
- Delete the AMICI files created in the GEARS results folder.
- Rerun the GEARS procedure for the model.

9.2 The results summary plot is not included in the figure report (Linux)

One minor problem that will not stop the GEARS procedure but is a small restriction is that the figure report created in Linux will not include the results summary figure. This is due to a Matlab restriction regarding the printing of uitables into eps format.

9.2.1 Warning

This problem will not output an error in GEARS but will output a warning. The results summary will still be saved as a fig file in the results folder defined by the user.

9.2.2 Solution

Unfortunately, as far as we are aware this is a Matlab restriction in Linux. If in Linux the figure will simply not be included in the figure report but not stop the GEARS procedure. As this is the only figure that contains a uitable this issue should not affect other figures created by GEARS.

9.3 Exportation to xls can result in decimal shifts depending on the decimal separators used.

With the exportation to xls format, some issues have been seen for users using commas as decimal separators. All of the correct results will still be saved in all other locations, this is only a problem with the xls exportation. Please be aware of this issues if you are using comma decimal separators and compare the results in the xls export to the other saved results.

9.4 The bounding procedure can potentially fail in the presence of a lack of identifiability.

For systems with identifiability issues the `GEARS` parameter bounding procedure can fail resulting in the regularised estimation finding local solutions.

9.4.1 Error

This will cause the regularised estimation to be in a local solution but not in the non-regularised estimation. This problem is particularly problematic for the Goodwin Oscillator problem. This can also be caused by the local solver hitting parameter bounds.

If the solution below fails then the following warning will appear.

```
The parameter bounding has failed after 5 corrective steps, bounds
will be set to the original values.  Please try increasing the
computational budget given to the solver.
```

9.4.2 Solution

As of `GEARS` V1.2 a quality of fit check has been added to the regularised estimation. This will increase the size of the parameter bounds to try and ensure that the regularised solution is included within the reduced parameter bounds.

If this robustness step fails please try increasing the computational budget given to `GEARS`.

We have also implemented a warning that will indicate if the local solver has hit the parameter bounds, this will be found in the warning log in the html report and the results structure. Please note that this warning can simply mean that a high-cost local search has hit the bounds, which would have minimal affect on the final result so more user investigation into the issue may be required.

10 Acknowledgements

This research has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 675585 (MSCA ITN "SyMBioSys") and from the Spanish MINECO/FEDER project SYNBIOCONTROL (DPI2017-82896-C2-2-R). Jake Alan Pitt is a Marie Skłodowska-Curie Early Stage Researcher at IIM-CSIC (Vigo, Spain) under the supervision of Prof Julio R. Banga.

11 Contact and feedback

Please check the document and examples carefully before contacting the authors. Please pay special attention to section 9 of this document. If needed, they can be contacted by email at jp00191.su@gmail.com and julio@iim.csic.es.

References

Egea, J. A., Henriques, D., Cokelaer, T., Villaverde, A. F., MacNamara, A., Danciu, D.-P., Banga, J. R., and Saez-Rodriguez, J. (2014). Meigo: an open-source software suite based on metaheuristics for global optimization in systems biology and bioinformatics. *BMC Bioinformatics*, 15(1):136.

Fröhlich, F., Theis, F. J., Rädler, J. O., and Hasenauer, J. (2016). Parameter estimation for dynamical systems with discrete events and logical operations. *Bioinformatics*, 33(7):1049–1056.

Gábor, A., Villaverde, A. F., and Banga, J. R. (2017). Parameter identifiability analysis and visualization in large-scale kinetic models of biosystems. *BMC Systems Biology*.

Villaverde, A. F., Barreiro, A., and Papachristodoulou, A. (2016). Structural identifiability of dynamic systems biology models. *PLoS Computational Biology*, 12(10):e1005153.