Genetic regulatory networks inference : modeling, parameters estimation & model validation

Fomekong Nanfack, Y.

Publication date
2010

2

Parameter estimation for biological systems[1]

Parameter estimation in systems biology is usually part of an iterative process to develop data-driven models for biological systems that should have predictive value. In this chapter we discuss how to obtain parameters for mathematical models by data fitting. We restrict ourselves to the case where a deterministic model in the form of a mathematical function-based model is available, say a system of differential and algebraic equations. For instance in the case of a biochemical process, hypotheses based on the knowledge of the underlying network structure of a pathway are translated into a system of kinetic equations, parameters are obtained from literature or estimated from a data fit, and with the resulting model predictions are made that can be tested with further experiments. To compare model results to experimental data one first has to simulate the mathematical model to produce these results, the *forward problem*. The *inverse problem* is the problem at hand: the estimation of parameters in a mathematical model from measured observations. There are a number of difficulties involved (see, e.g., [238]). The forward problem requires a fast and robust time integrator. Fast, because the model will be evaluated many times. Robust, because the whole parameter and state space will be visited, which most likely will result in a different character of the mathematical model (number and range of time scales involved). The inverse problem has even more pitfalls. The first question is whether the parameters for the mathematical model can be determined assuming that for all observables continuous and error-free data are available. This is the subject of *a priori identifiability* or

[1]This chapter is based on the paper Maksat Ashyraliyev and Yves Fomekong-Nanfack and Jaap A. Kaandorp and Joke Blom, "Systems biology: parameter estimation for biochemical models", FEBS Journal, 276(4):886 - 902(17). 2009. [5]

structural identifiability analysis of the mathematical model. The actual parameter estimation or data fitting typically starts with a guess about parameter values and then changes those values to minimize the discrepancy between model and data using a particular metric. Kinetic models with nonlinear rate equations have in general multiple sets of parameters that lead to such minimisation, some of those minima may only be local. The value of parameters and model variables may range over many orders of magnitude, one can get stuck in a local minimum or one can wander around in a very flat part of the solution space. Given a particular set of experimental data and one particular acceptable model parameterization obtained by a parameter estimation procedure does not mean that all obtained parameters can be trusted. After the minimum has been found, an *a posteriori* or *practical identifiability* study can show how well the parameter vector has been determined given a data set that is possibly sparse and noisy. That this part of model fitting should not be underestimated is shown by Gutenkunst et al. [103]. For all 17 systems biology models they considered, the obtained parameters are "sloppy", meaning not well-defined. On the other hand, one could argue that often the precise value of a parameter is not required to draw biological conclusions [6].

This chapter discusses the different main issues and approaches encountered in parameter estimation problem of model of biological system. First, a problem definition is given followed by the choice of measure for the goodness of the fit. Second, we discuss different methods for both the a priori and a posteriori identifiability. Next, we give a brief survey of the current methods used in parameter estimation with a focus on those that are implemented in popular toolboxes for systems biology. In Section 2.4 we give some guidelines on the application of these methods in practice. Finally, in the Appendix A 8.4 an overview is given of the contents of some well-known toolboxes.

2.1 Problem definition

Deterministic models arising from kinetic equations are typically given by a system of differential algebraic equations (DAEs)[1] - ordinary differential equations (ODEs) coupled to algebraic equations - of the form:

$$\begin{cases} A\frac{\mathrm{d}\mathbf{x}(t,\boldsymbol{\theta})}{\mathrm{d}t} = \mathbf{f}(t,\mathbf{x}(t,\boldsymbol{\theta}),\boldsymbol{\theta},\mathbf{u}(t)), \quad t_0 < t \leq t_e, \\ \mathbf{x}(t_0,\boldsymbol{\theta}) = \mathbf{x}_0(\boldsymbol{\theta}), \end{cases} \tag{2.1}$$

where t denotes time, the m-dimensional vector $\boldsymbol{\theta}$ contains all unknown *parameters*, $\mathbf{x}$ is an n-dimensional vector with the *state variables* (e.g., concentration values), $\mathbf{u}$ are the externally *input* signals, and $\mathbf{f}$ is a given vector function.

[1] The content of this chapter is also applicable to (discretized) systems of partial differential equations (PDEs) and delay differential equations (DDEs). Fitting parameters of stochastic models requires a different approach (see e.g., [97, 225, 272]).

When components of the initial state vector $\mathbf{x}_0$ are not known, they are considered as unknown parameters, so $\mathbf{x}_0$ may depend on $\boldsymbol{\theta}$. In the simplest case, A is a constant diagonal $n \times n$ matrix with $A_{ii} = 1$ if the i-th equation is a differential equation and $A_{ii} = 0$ if the i-th equation is algebraic.

In addition, a vector of *observables* is given

$$\mathbf{g}(t, \mathbf{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta}, \mathbf{u}(t)), \tag{2.2}$$

which are quantities in the model - in general (a combination of) state variables - that can be experimentally measured, and possibly a vector of (non)linear *constraints*

$$\mathbf{c}(t, \mathbf{x}(t, \boldsymbol{\theta}), \boldsymbol{\theta}, \mathbf{u}(t)) \geq 0. \tag{2.3}$$

Let us assume that N measurements are available to find parameters of system (2.1-2.3). Each measurement, which we denote by y_i, is specified by the time t_i when the i-th component of the observable vector $\mathbf{g}$ is measured. The corresponding model value for a specific parameter vector $\hat{\boldsymbol{p}}$, which can be obtained sufficiently accurate by numerical integration of system (2.1) and computing the observable function (2.2), is denoted by $\hat{g}_i = g_i(t_i, \mathbf{x}, \hat{\boldsymbol{\theta}}, \mathbf{u})$. The vector of discrepancies between the model values and the experimental values is then given by $\mathbf{e}(\hat{\boldsymbol{p}}) = |\mathbf{g}(t, \mathbf{x}(t, \hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}}, \mathbf{u}(t)) - \mathbf{y}|$. We assume that system (2.1) is a sufficiently accurate mathematical description approximating reality. This means that all relevant knowledge about the biological processes is incorporated correctly in the vector function $\mathbf{f}$. Thus, the only uncertainty in (2.1) is the vector of unknown parameters $\boldsymbol{\theta}$. In this case the difference $e_i(\boldsymbol{\theta}^*) = |g_i(t_i, \mathbf{x}, \boldsymbol{\theta}^*, \mathbf{u}) - y_i|$ is solely due to experimental errors, where $\boldsymbol{\theta}^*$ is the true solution.

2.1.1 Fitness criterion

The m-dimensional optimization problem is given by the task to minimize some measure, $V(\boldsymbol{\theta})$, for the discrepancy $\mathbf{e}(\boldsymbol{\theta})$. By far the most used measure for the discrepancy is the Euclidean norm or the sum of the squares weighted with the error in the measurement

$$V_{MLE}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \frac{(g_i(t_i, \mathbf{x}, \boldsymbol{\theta}, \mathbf{u}) - y_i)^2}{\sigma_i^2} = \mathbf{e}^T(\boldsymbol{\theta}) W \mathbf{e}(\boldsymbol{\theta}), \tag{2.4}$$

see [63, 91]. This measure results from the Maximum Likelihood Estimator (MLE) theory. Under the assumption that the experimental errors are independent and normally distributed with standard deviation σ_i, the least squares estimate $\hat{\boldsymbol{\theta}}$ of the parameters is the value of $\boldsymbol{\theta}$ that minimizes the sum of squares

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} V_{MLE}(\boldsymbol{\theta}). \tag{2.5}$$

When these assumptions do not hold, other measures than $V_{MLE}(\boldsymbol{\theta})$ might be used like the sum of the absolute values. The MLE theory then does not apply so $\hat{\boldsymbol{\theta}}$ is not the least squares estimate and the statistical analysis of Section 2.3.1.2 does not hold. Dependent on the optimization method or the mathematical discipline the function $V(\boldsymbol{\theta})$ is called *objective* function, *cost* function, *goal* function, *energy* function or *fitness* function.

The solution set $\hat{\boldsymbol{\theta}}$ in Equation 2.5 may be a unique point, a countable (finite or infinite) collection of points, or a set containing an uncountable number of points. The following three examples illustrate these types of solution sets $\hat{\boldsymbol{\theta}}$.

Example 2.1.1 *: $\hat{\boldsymbol{\theta}}$ contains unique solution. Suppose that $V(\boldsymbol{\theta}) = \boldsymbol{\theta}^T\boldsymbol{\theta}$ and $\boldsymbol{\theta} = \mathbb{R}^m$ (i.e. $\boldsymbol{\theta}$ is unconstrained). The value $\boldsymbol{\theta} = 0$ uniquely minimises V. Hence, $\boldsymbol{\theta}^*$ is the single point $\boldsymbol{\theta}^* = 0$.*

Example 2.1.2 *: $\hat{\boldsymbol{\theta}}$ has countable (finite or infinite) number of points. Let $\boldsymbol{\theta}$ be a scalar and $V(\boldsymbol{\theta}) = \sin(\boldsymbol{\theta})$. If $\boldsymbol{\theta} = [0, 4\pi]$, then $\sin(\boldsymbol{\theta}) = -1$ (its minimum) at the point $\boldsymbol{\theta}^* = \{3\pi/2, 7\pi/2\}$, a countable set with a finite number (two) of elements. On the other hand, if $\boldsymbol{\theta} = \mathbf{R^1}$, then $\boldsymbol{\theta}^* = \{..., -5\pi/2, -\pi/2, 3\pi/2, 7\pi/2, ...\}$, a countable set with infinite number of elements.*

Example 2.1.3 *: $\hat{\boldsymbol{\theta}}$ has uncountable number of points. Suppose that $V(\boldsymbol{\theta}) = (\boldsymbol{\theta}^T\boldsymbol{\theta} - 1)^2$ and $\boldsymbol{\theta} = \mathbf{R}^m$. This cost function is minimised when $\boldsymbol{\theta}^T\boldsymbol{\theta} = 1$. which is the set of points lying on the surface of a* m*-dimensional sphere having radius 1. When $m \geq 2$, $\boldsymbol{\theta}^*$ is an uncountable (but bounded) set.*

2.2 Parameter estimation methods

To find the minimum of the objective function, optimization methods are used. We describe here two classes: *local* and *global*. Local search methods typically converge fast to a minimum, but as the name suggests, this might be a local minimum and the method has no possibility to escape from this minimum to find the true or global minimum. For local search methods there is in general a theoretical proof of convergence (and of convergence speed) to the minimum *if* the initial guess is sufficiently close to that minimum. Global optimization searches all over the parameter space to find smaller and smaller values for the objective function, but in general there is no proof for convergence to *the* minimum (with exception of the simulated annealing algorithm).

Various numerical algorithms exist for global and local optimization. In [175, 184] a number of global and local methods are applied to a benchmark of biochemical pathways. Below we will describe briefly the methods that are frequently used when estimating model parameters of biological problems and methods that are available in general toolboxes used in system biology.

2.2.1 Some definitions and theorems

$\hat{\boldsymbol{\theta}}$ is a *global minimizer* of the objective function V if it gives the lowest obtainable objective function value from an *arbitrary starting point*:

$$\hat{\boldsymbol{\theta}}_{global} = \arg\min_{\boldsymbol{\theta}} V(\boldsymbol{\theta}), \qquad \forall \boldsymbol{\theta} \text{ in the parameter space.} \tag{2.6}$$

$\hat{\boldsymbol{\theta}}$ is a *local minimizer* of the objective function V if it gives the lowest obtainable objective function value in the *neighbourhood of the starting point*:

$$\hat{\boldsymbol{\theta}}_{local} = \arg\min_{\boldsymbol{\theta}} V(\boldsymbol{\theta}), \qquad \forall \|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_{start}\| < \delta, \quad \delta > 0. \tag{2.7}$$

A *stationary point* x^* of a function f is a point for which the gradient is zero

$$\nabla f(x^*) = 0. \tag{2.8}$$

The following theorems hold for unconstrained optimization and a sufficiently differentiable objective function V. In this case V can be extended into a Taylor series around $\hat{\boldsymbol{\theta}}$:

$$V(\hat{\boldsymbol{\theta}} + \delta\boldsymbol{\theta}) = V(\hat{\boldsymbol{\theta}}) + \delta\boldsymbol{\theta}^T \nabla V(\hat{\boldsymbol{\theta}}) + \frac{1}{2}\delta\boldsymbol{\theta}^T \nabla^2 V(\hat{\boldsymbol{\theta}})\delta\boldsymbol{\theta} + \ldots \tag{2.9}$$

with the gradient

$$\nabla V(\hat{\boldsymbol{\theta}}) = [\frac{\partial V}{\partial \boldsymbol{\theta}}(\hat{\boldsymbol{\theta}})], \tag{2.10}$$

and the Hessian or second derivative

$$\nabla^2 V(\hat{\boldsymbol{\theta}}) = [\frac{\partial V}{\partial \boldsymbol{\theta}_i \partial \boldsymbol{\theta}_j}(\hat{\boldsymbol{\theta}})]. \tag{2.11}$$

A *necessary* condition for a parameter vector $\hat{\boldsymbol{\theta}}$ to be a *local minimizer* of V is that $\hat{\boldsymbol{\theta}}$ is a *stationary point* of V:

$$\nabla V(\hat{\boldsymbol{\theta}}) = 0.$$

A *sufficient* condition for a *local minimizer* is that $\hat{\boldsymbol{\theta}}$ is a *stationary point* of V and the *Hessian of V is positive definite*:

$$\nabla V(\hat{\boldsymbol{\theta}}) = 0; \qquad \boldsymbol{\theta}^T \nabla^2 V(\hat{\boldsymbol{\theta}})\boldsymbol{\theta} > 0 \quad \forall \boldsymbol{\theta} \neq 0.$$

2.2.2 Global Optimization

Most global optimization methods are stochastic of nature to prevent the search process being trapped in a local minimum. Moles et al. [184] have performed a comparison of a number of global optimization methods on parameter estimation problems for biochemical pathways.

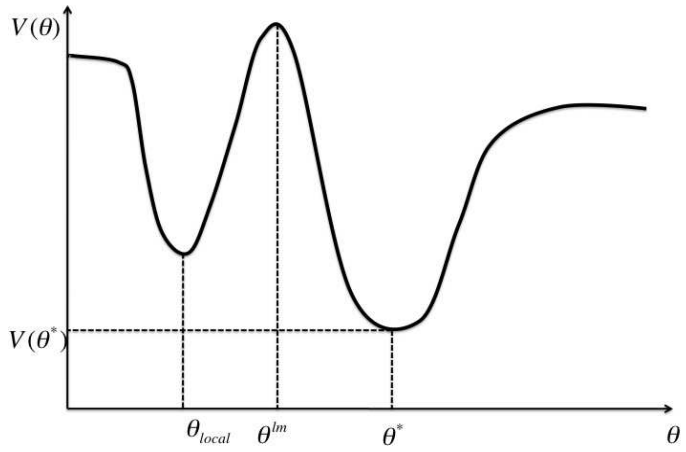

Figure 2.1: In this simple example, the landscape shows two minima: a local minima at θ_{local} and a global minima at θ^*. The standard search optimization techniques available in the literature for solving the parameter estimation problem can be classified into 2 categories. The local search methods and the global search method. If the initial guess is around this local minima, the algorithm will converge to the global minimum. For a given starting point, the local search method does not necessary find the global minima since one of the major drawback is their incapacity to get out of local minima. For instance, any starting point for wich $\theta \leq \theta^{lm}$, most local search methods will be trapped at $\theta local$. Global methods offer an interesting alternative when no initial guess is available, since the can overcome this problem.

2.2.2.1 Simulated Annealing

Simulated Annealing (SA) is a stochastic optimization algorithm proposed by Kirkpatrick et al. [140] in 1983. The term *annealing* comes from physics. It is the process of heating up a solid until it melts, followed by a slow cooling down until the molecules are aligned in a crystalline structure corresponding to the minimum energy state. The cooling must occur at a sufficiently slow rate, otherwise the system will end up in an amorphous or polycrystalline state and thus the system will not be at its minimum energy state. In optimization, the SA algorithm attempts to mathematically capture the process of controlled cooling associated with physical processes; the analogy to the minimum energy state is the minimum value for the objective function.

SA is based on the Metropolis algorithm [177] which is a Monte-Carlo method to sample a thermodynamic system. Rephrased for the parameter estimation problem it samples for a fixed "temperature" the parameter space according to the Boltzmann-Gibbs probability distribution

$$P(\boldsymbol{\theta}) = C \exp\left(-\frac{V(\boldsymbol{\theta})}{k_B T}\right), \tag{2.12}$$

where C is a normalization constant, k_B the Boltzmann constant, and T the temperature. Starting from an initial (random) parameter vector, in each step a random new state (parameter vector) is generated based on the previous one. This new state is accepted with a certain probability (see below under Transition probability). If it is rejected a new state is generated based on the same parameter vector as before. In this way a Markov chain is obtained which, if it is sufficiently long, describes the required probability distribution. The macroscopic observable, the minimizing parameter vector, is the average over all states in the Markov chain. In SA the Metropolis algorithm is applied with a slowly decreasing T. SA starts with a high "temperature" implying that all states, or parameter vectors, are equally probable. The original algorithm - the *homogeneous* Markov chain method - computes for a constant temperature a complete Markov chain, i.e. the required probability distribution is obtained. Then the temperature is slowly decreased and the next distribution is sampled. In contrast the *inhomogeneous* Markov chain method decreases the temperature every time a new state has been found. Devising the cooling schedule - initial temperature, method of lowering the temperature, and the stop criterion - is the art of simulated annealing. Under certain conditions (ergodicity, cooling schedule) it has been proven that SA converges to the global minimum (see, e.g., [278]).

Cooling schedules Many have attempted to derive theoretical or experimental proofs of an efficient cooling schedule scheme (see [24] for an extensive review). Among the most popular ones, three different theoretical concepts are used

Logarithmic: Introduced by Geman and Geman [90], this one has special theoretical importance. The temperature is decreased according to: $t_i = \gamma/log(i + d)$ with i the iteration count and d usually set to one. Although it has been proven that for $\gamma \geq E_{max}$, the true global minima can be found (in the limit of infinite time), with E_{max} being the maximum energy barrier (problem dependent and a priori unknown), this method is very slow and impractical because of its asymptotically slow temperature decrease [105].

Geometric: The original cooling schedule proposed by Kirkpatrick [140] and still widely used with major or minor variants. The temperature is updated by: $t_i = \alpha t_{i-1}$. The cooling factor α is assumed to be a constant smaller than one. Examples of usage and a good explanation of the underlying mechanisms are given by Johnson [127].

Adaptive: The previous cooling schedules always apply the same cooling factor irrespective the state of the system. It is known that at high temperature, almost all new parameter vectors are accepted although some of

them are bad solutions. It is obvious that using an appropriate cooling schedule depending on the state of the system can lead to large improvements. A variety of adaptive temperature annealing strategies have been proposed. The main techniques are presented by Boese [24]. The most important ones are: *(i)* Lam [155,156]: the temperature is updated aiming to maintain the system in thermodynamical equilibrium; *(ii)* Ingber [116,118]: a very popular cooling schedule. The strength of this algorithm is that it takes into account the sensitivity of the cost function for each parameter. The goal is to extend the insensitive parameter's search range relative to the range over the more sensitive parameters. Each parameter has its own temperature, equally initialized at the beginning. After every N_{acc} accepted steps, the sensitivity for the best solution parameters is computed and after every N_{gen} generation steps the temperatures are re-annealed scaled by the sensitivities. A very limited number of method parameters has to be assigned by the user: the rate control parameter C, N_{acc}, and N_{gen}. The other method parameters are automatically set and updated by the algorithm. The optimal values of the three parameters are problem dependent [117], but the performance of the algorithm is not critically influenced for choices of C in the range of 1 to 10, N_{acc} O(10-100) and N_{gen} O(200-1000).

Transition probability If the objective function of the new parameter vector $\boldsymbol{\theta}'$ is smaller than the previous one then the new parameter vector is accepted. However, to prevent getting stuck in a local minimum, the new parameter vector is also accepted with a probability according to the sampled distribution

$$P(\Delta V, T) = exp\left(-\frac{\Delta V}{k_B T}\right), \qquad \text{with } \Delta V = V(\boldsymbol{\theta}') - V(\boldsymbol{\theta}). \tag{2.13}$$

Expression (2.13) is known as the *Metropolis Criterion*. For $T \to 0$ and $\delta V > 0$, the probability $P(\delta V, T) \to 0$. Therefore, for sufficiently small values T, the process will more and more go "downhill": new accepted parameter vectors tend to have lower objective function values.

2.2.2.2 Evolutionary Algorithms

Evolutionary Algorithms (EA) are inspired by biological evolution. Potential solutions (parameter vectors) are the *individuals* of a *population*. In order to get new solutions - a next *generation* - the individuals in the population are replaced using mechanisms as *reproduction*, *natural selection*, *mutation*, *recombination*, and *survival of the fittest*.
Initially, a population is created of random individuals (possible parameter vectors). Next, the corresponding objective functions are computed which define the fitness of an individual (the higher the fitness, the better the solution). The selection process is mimicked by assigning probabilities to individuals related to their fitness to indicate the chance to get selected for the next generation. Individuals with a high fitness are assigned high probabilities. New

individuals are created by two operators: recombination (or cross-over) and mutation. Recombination consists in selecting some parents (at least two) and results in one or more children (new candidates). Mutation acts on one candidate and results in a new candidate. These operators create the offspring (a set of new candidates). These new candidates compete with old candidates for their place in the next generation (survival of the fittest). This process can be repeated until a candidate with sufficient quality (a solution) is found or a predefined computational limit is reached. There are many different ways of writing these operators and one can find exhaustive literature focussing on this aspect of EAs (see, e.g. [96]).

EA operators
The selection operator is responsible for convergence to the minimum, the recombination operator for exploring the parameter space and the mutation operator gives nearby solutions a chance to survive.

Fitness A commonly used objective-to-fitness transformation results in a fitness value of $\max(0, C_{max} - V(\boldsymbol{\theta}))$ with C_{max} either a user-defined constant or the maximum V-value thus far. To prevent almost equal selection probabilities in later stages of the algorithm the fitness values should be scaled accordingly [96]. Another transformation is simply rank-based, where the population is sorted according to their objective values and fitness assignment depends only on the position [11,287].

Selection determines, which individuals are chosen for mating (recombination) and how many offspring each selected individual produces. The first step is fitness assignment. Next, the actual selection is performed. Parents are selected according to their fitness by means of one of the following algorithms [96]:
truncation: the only deterministic selection: select the m best individuals and reproduce them until the pool is filled;
roulette-wheel: selection with size of wheel part proportional to fitness [13];
stochastic remainder: sampling. First $\mathbf{entier}(f_i/\bar{f})$[2] times individual i are selected with f_i the individual and $\bar{f}$ the average fitness. Next the pool is filled using a weighted toss [13];
tournament: N "tournaments" will be held with K randomly picked individuals as competitors for a place in the pool. Winner is the one with highest fitness [179].
The selection process is an extremely important part of the convergence of the algorithm: if the selection pressure is high - as with *roulette-wheel* - then the convergence time is fast, but the solution can be a local one. If the selection pressure is low - as with *tournament* with small K - it is the other way around.

[2] $\mathbf{entier}(x)$ is the largest integer value not exceeding x.

Recombination or cross-over produces new individuals by combining the information contained in the parents (parents: mating population). In the case of real-valued variables the algorithms all choose a point on the line connecting the two parents, either deterministically -*line recombination* (interpolation with a fixed constant) - or stochastically. In the latter case one distinguishes *intermediate recombination* in which a point is chosen in an interval slightly larger than the connecting line segment and *extended line recombination* where the complete line is used but the probability decreases with the distance from a parent.

Mutation consists in randomly altering an individual. The mutation step (usually very small) is the probability of mutating a variable, and the mutation rate is the effective mutation applied to that variable. Although in general the mutation step is inversely proportional to the dimension of the problem, the mutation rate does not depend on the problem.

Reinsertion ("survival of the fittest") After producing offspring they must be inserted into the population. This is especially important, if the number of offspring does not equal the size of the original population. To guarantee that the best individual(s) survive the *elitist strategy* [96] can be used.

2.2.2.3 Covering methods

Covering methods are deterministic global optimization algorithms, that guarantee that a solution with a given accuracy is obtained. The price paid for this guarantee, however, is that some a priori information of the function must be available.

Branch and Bound (B&B) [157, 181] requires that the search space is finite (parameters are constrained) and can be divided to create smaller subspaces. To apply branch and bound, one must have a means of computing upper and lower estimated bounds of the objective function to be minimized.

The method starts by considering the original problem with the complete search space - the *root problem*. The lower-bounding and upper-bounding procedures are applied to the root problem. If the bounds match, then an optimal solution has been found and the procedure terminates. Otherwise, the search space is partitioned into two or more regions. These subproblems become children of the root search node. The algorithm is applied recursively to the subproblems, generating a tree of sub-problems. If an optimal solution is found to a subproblem, it is a feasible solution to the full problem, but not necessarily globally optimal. Since it is feasible, it can be used to prune the rest of the tree: if the lower bound for a node exceeds the best known feasible solution, no globally optimal solution can exist in the subspace of the feasible region represented by the node. Therefore, the node can be removed from consideration. The search proceeds until all nodes have been solved or pruned, or until some specified

threshold is met between the best solution found and the lower bounds on all unsolved subproblems. Although this method is widely used in engineering, the technique is not that popular among the biologists and computational biology community.

2.2.3 Local optimization

If the gradient of the objective function can be computed one can solve the minimization problem by finding the point where the gradient vanishes using *gradient-based methods*. *Direct search methods* try to find the minimising point of the objective function without explicitly using derivatives. As for the global search methods these methods only require an order relation ($V(\boldsymbol{\theta}_1) < V(\boldsymbol{\theta}_2)$) for all points in parameter space.

2.2.3.1 Direct-search methods

The term direct-search method has first been used in 1961 in the classical paper of Hooke and Jeeves [111] that describes their pattern search method, but it is more generally used for all methods that find a local minimum without the use of a derivative. Direct search methods select a finite - generally not large - number of possibilities each step and check whether one of these is better than the current one. For reviews on direct-search or derivative-free methods we refer to [147, 212, 293]. Here we discuss the two most used methods: the classical *Hooke-Jeeves* method [111] and the *Nelder-Mead* or *Downhill Simplex* method [194].

Hooke-Jeeves method The pattern search method of Hooke and Jeeves [111] consists of two steps. In the first a series of *exploratory* changes of the current parameter vector are made, typically a positive and negative perturbation of one parameter at a time. The exploratory step then has formed a basis for the parameter space with information in which directions the objective function decreases. In the next step, the *pattern move*, the information obtained is used to find the best direction for the minimisation process. The original method is a special case of *generalized pattern search* methods for which it is shown that the search directions span the parameter space [274]. For a nice discussion on this type of direct-search methods, the broad class of *generating set search* methods, including convergence results, some history and references to other ideas we refer to the extensive review paper of Kolda et al. [147]. They show amongst others that these methods have the same type of convergence guarantee as gradient-based methods.

Nelder-Mead simplex algorithm The Nelder-Mead method [153,194] is based on the idea of an adaptive simplex - the simplest polytope of $m + 1$ vertices in m dimensions (2D: triangle, 3D: tetrahedron). The objective function is evaluated in all vertices ($\boldsymbol{\theta}$"s) and the vertices are ordered according to the value.

The next step tries to replace the "worst" vertex by a better one. A line search is performed along the line through this vertex and the centroid of the remaining vertices: $\boldsymbol{\theta}_{new} = \boldsymbol{\theta} + \alpha\boldsymbol{\theta}_{worst}$. For $\alpha = 1, 2, \frac{1}{2}, -\frac{1}{2}$ it is tested whether the new objective value is better than the old one. If this is the case the simplex is adapted by replacing the old vertex by the new one. If not, a *shrink* procedure is performed: the "best" vertex stays in the simplex, all other ones are replaced by a vertex half-way along the line from the best vertex. If the line search is successful the method uses just 1 to 4 function evaluations per step and the aim is that the simplex adapts itself to the minimising function. But in contrast to the Hooke-Jeeves method it improves the objective function value along the sequence of worst vertices.

2.2.3.2 Gradient-based methods

In constrast to all other methods above this class of methods not only requires the value of the objective function but also of its first derivative with respect to the parameters. These type of methods are not so straightforward to implement as the direct-search methods, but if it is possible to use them it is in general preferable to do so. Often in implementations approximations of the gradient and/or the Hessian (second derivative) are used, e.g., by finite differences. However, with the current automatic differentation tools like ADIFOR [22], symbolic algebra packages like Maple [88] and Mathematica [227], and modelling languages with automatic computation of derivatives like AMPL [82] and GAMS [130], it is doable and preferrable to use the exact derivative. For a general treatment of this subject we refer to Nocedal [195].

Remember that a requirement for a local minimizer $\boldsymbol{\theta}^*$ is that the gradient $\nabla V(\boldsymbol{\theta}^*) = 0$ (stationary point). A sufficient condition requires that the Hessian is positive definite. Note that none of the methods below guarantees the latter requirement!

Gradient-based methods are all *descent methods*. These methods first find a descent direction $d\boldsymbol{\theta}$ and then take a step $\alpha d\boldsymbol{\theta}$ in that direction, with α such that it results in a "good" decrease of the objective function

$$\boldsymbol{\theta}_{new} = \boldsymbol{\theta} + \alpha d\boldsymbol{\theta}, \qquad V(\boldsymbol{\theta}_{new}) < V(\boldsymbol{\theta}). \tag{2.14}$$

The largest gain is obviously obtained when α is determined by a line-search, i.e., by finding the minimum value of $V(\boldsymbol{\theta}+\alpha d\boldsymbol{\theta})$ for all $\alpha > 0$. Note that a simple decrease in the objective function ($f(x_{k+1}) < f(x_k)$) is not sufficient to converge to a stationary point of f. (Counterexample: $V(x) = x^2$ and $x_i = 1 + 2^{-i}$, see [57].)

Steepest descent or gradient method In this method the search direction is defined by the gradient

$$d\boldsymbol{\theta} = -\nabla V(\boldsymbol{\theta}). \tag{2.15}$$

In the final stage, however, this method has a slow convergence. In fact if combined with exact line search it can even fail.

Newton's method Newton's method iteratively solves the equation for a stationary point $\nabla V(\boldsymbol{\theta}^*) = 0$ by linearization. The search direction for the line-search method is in this case

$$d\boldsymbol{\theta} = -\nabla^{-2} V(\boldsymbol{\theta}) \nabla V(\boldsymbol{\theta}). \tag{2.16}$$

In quasi-Newton methods the Hessian is approximated. If the starting point is sufficiently close to the solution Newton's method has a quadratic order of convergence.

Trust region method [47] The objective function $V(\boldsymbol{\theta})$ is approximated by a simpler function which mimicks the behaviour of V in a neighbourhood of $\boldsymbol{\theta}$. This function is then minimized over this neighbourhood, the *trust region*, and if the objective function decreases the new value is accepted. Otherwise the trust region is decreased. Originally the approximation consisted of the first two terms of the Taylor expansion of V at $\boldsymbol{\theta}$ but for high-dimensional problems this is still too expensive. In this case the trust region is restricted to two dimensions [34]. This subspace is spanned by the gradient vector ∇V (Equation (2.15)) and a direction of negative curvature given by $d\boldsymbol{\theta}^T \nabla^2 V(\boldsymbol{\theta}) d\boldsymbol{\theta} < 0$ or the Newton direction (Equation (2.16)). The aim of the first combination is global convergence and of the second fast local convergence.

2.2.3.3 Gradient-based methods for least-squares

Gauss-Newton If the function to be minimized is a sum of squares - as is the case when solving a least-squares problem - Newton's method is often replaced by a modification - the Gauss-Newton algorithm - in which the Hessian is not used. The gradient of $V_{MLE}(\boldsymbol{\theta}) = \mathbf{e}^T\mathbf{e}$ is given by $\nabla V_{MLE} = J^T\mathbf{e}$, where the Jacobian $J(\boldsymbol{\theta}) = \frac{\partial \mathbf{e}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta})$ is the so-called 'sensitivity" matrix of size $N \times m$ (cf. Equation (2.19)).
To solve for the stationary point again linearization is used which results in the task to solve the *normal equations*

$$J^T(\boldsymbol{\theta}) J(\boldsymbol{\theta}) \delta\boldsymbol{\theta} = -J^T(\boldsymbol{\theta}) \mathbf{e}(\boldsymbol{\theta}). \tag{2.17}$$

Note that $\delta\boldsymbol{\theta}$ is a descent direction because $\delta\boldsymbol{\theta}^T \nabla V_{MLE} = \delta\boldsymbol{\theta}^T J^T \mathbf{e} = -\delta\boldsymbol{\theta}^T J^T J \delta\boldsymbol{\theta} < 0$. As in Newton this is an iterative process.

Levenberg-Marquardt method [171] can be seen as Gauss-Newton with damping or as a combination of Gauss-Newton with steepest descent. The search direction is defined by

$$\left(J^T(\boldsymbol{\theta})J(\boldsymbol{\theta}) + \lambda I_m\right)\delta\boldsymbol{\theta} = -J^T(\boldsymbol{\theta})\mathrm{e}(\boldsymbol{\theta}), \tag{2.18}$$

where $\lambda \geq 0$ is some constant and I_m the identity matrix of size m. $\delta\boldsymbol{\theta}$ is a descent direction for all $\lambda > 0$; for λ large Equation (2.18) results in the steepest descent method and for λ small in the Gauss-Newton process. The first is a good strategy in the initial stage of the process, the latter in the final stages. The art of the Levenberg-Marquardt method is the design of the damping factor λ (see, e.g., [33, 165]).

2.2.4 Hybrid methods

Global methods in generally work well to explore the parameter space but are slow in finding the minimum of the objective function precisely (see, e.g. [78]). In contrast, local methods are much faster in finding a minimum *once in the neighbourhood*. Sequential application of both approaches combines the best of the two. Such hybrid methods use a global search method to identify promising regions of the search space that are further explored by a local optimizer.

Although hybrid methods are quite recent in the systems biology world, these methods have a long tradition in computational problems [281]. Katare et al. [132, 133] employ a Particle Swarm Optimization [136, 137] combined with Levenberg-Marquardt. However, their method appears to be sensitive to the "swarm topology" that defines the information transfer between the parameter vectors. Combinations of local search with the Stochastic Ranking Evolution Strategy [233] seem to be more promising. In [228], Rodriguez-Fernandez et al. applied with good results SRES + DN2GB (Gauss-Newton + trust region for stabilization) on the three-step pathway benchmark problem [184]. In [6, 78] a challenging reaction-diffusion system is considered describing the early *Drosophila* development. This results in a model with 348 state variables and a 66-dimensional optimisation problem with (non)linear constraints. Jaeger et al. [121] obtained previously the parameters for that model with parallel simulated annealing. Fomekong Nanfack et al. [78] show that the hybrid method SRES+Nelder-Mead is approximately 50 times as fast. The same problem was solved with SRES+DS+LM [6] with a comparable speed up but a better approximation of the local minima.

Another interesting approach is an intrinsic global-local method such as the scatter-search method [154, 228], an evolutionary algorithm with a local search method after (each) recombination step. Since this method is expensive for costly objective funtion evaluations SSKm (Scatter-search-Kriging) [66] has been developed. Here the number of "local-search" points is reduced by predicting the possibility that a new parameter vector will result in a lower min-

imum without evaluation of the objective function, based on the assumption that V has a Gaussian distribution (Kriging).

2.2.5 Constraints

For all optimization methods described above it holds that it is the implementation that counts, i.e. one version of an optimization method with different method parameters and strategy can result in a much better and faster convergence behaviour (for some problems) than the next. This holds even more for the implementation of constraints. Constraints can be implemented as penalties added to the objective function. This is often done in global and in direct search methods. It implies that the constraints are not strictly obeyed, at least during the search. In direct search methods linear constraints restrict the search directions -the parameter space becomes a cone - and thus the chance of failure increases (the search directions no longer span the search space). For nonlinear constraints a number of approaches exists (see Kolda et al. [147] for an overview of methods used in Generalized Set Search (GSS) methods). If the constraints are differentiable, this direction can be used when computing the new search direction. For GSS and gradient-based methods one can also solve an augmented nonlinear system where a Lagrange multiplier with the constraint is added and possibly other penalty terms [147, 257].

2.3 Model analysis and validation

As stated previously, the first aim of gene regulatory network inference consists in deriving the topology of a system. In our case, we focus on complex dynamical system exhibiting spatial temporal pattern formation. Beside the topology, the inference also aims at revealing the mechanistically details behind the system's dynamic. The mathematical models used, are necessarily simplified description of the phenomena being studied. The limited representation introduces uncertainty in the model, which is increased by the uncertainty inherited from the measured data. Furthermore, most GRNs are inferred from stochastic search algorithms. The stochastic nature of these algorithms imposes one to run as many as possible optimization trials on the same set of data before any deduction. From these many trials, different sets of parameter solutions can be obtained with reasonably good/low fitness.

In practice, most models based their correctness by visually comparing the simulated concentration of some molecular species with experimental data. The accumulation of the different sources of uncertainty imposes to further analyse the estimated parameters and the network inferred. More objective validation criteria are necessary to gain a certain level of confidence in the model's accuracy, the structural identifiability, the reliability of the parameters and overall the robustness of the model [124].

2.3.1 Identifiability

Unlike the simple motivational examples above given in Section 2.1.1, most biological problems have very complex form of cost function V for which solving Equation 2.5 analytically is impossible. The problem becomes harder for large-scale optimisation problem, hence requires optimisation approaches. Most Parameter estimation methods are iterative algorithms that start from an initial (random) parameter $\boldsymbol{\theta}$ and step by step improve the value to the closest possible true value $\boldsymbol{\theta}^*$ in order to minimise the cost function. The term "true value" assumes that for a given system of the form input-output data, a set of parameter has to be identified. It might not be possible to determine the true value of the parameter. Such a parameter is considered to be non-identifiable. The identifiable problem of a model is of great importance. It basically answears to one question: if one can not find the true parameters solution under ideal conditions (noise-free observations, error-free model paradigm and independently of the particular values of the parameters) , or if a model has more than one solution (i.e. more than one set of parameter values that will provide an identical fit to the data), then the physiological conclusions might be bias and conclusions could be different from solution to solution. Therefore, the model might not be relevant to prove anything about the biological system.
The *sensitivity matrix* J of the model is given by the sensitivity coefficients of the observables with respect to the parameters:

$$J = \left(\frac{\partial g_i(\boldsymbol{\theta})}{\partial p_j} \right). \tag{2.19}$$

A parameter is *globally identifiable* if it can be *uniquely* determined given the input profile $\mathbf{u}(t)$ and assuming continuous and error-free data for the observables of the model. If there is a *countable* number of solutions the parameter is *locally identifiable*; it is *unidentifiable* if there exist *uncountable many* solutions. A model is *structurally globally/locally identifiable* if all its parameters are globally/locally identifiable[3]

2.3.1.1 a priori identifiability

Before any further parameter estimation, once the model has been designed and the data are available, one should ask: **Given a model structure and an experimental protocol (ideal data), do the data contain enough information to estimate the unknown parameters of the model?** This is a theoretical question and not an easy one in general as most biological phenomena dealing with GRN describing a system dynamic in terms of mathematics model are very complex, especially if it is described as a nonlinear dynamic model.

[3]Note that these definitions are not always the same. Other definitions are: A model is *structurally identifiable* if its sensitivity matrix satisfies two conditions: each column has at least one large entry and the matrix has full rank [124]. A model is *locally identifiable* if it is *globally identifiable in a neigbourhood* of the parameter [228].

There are several techniques to determine a priori global identifiability of the model, but for realistic situations, i.e. non-linear models of a certain size, it is very difficult to obtain any results. Still it is advisable to always perform an a priori analysis, since parameter estimation methods can have problems with locally identifiable or unidentifiable systems. Symbolic algebra packages like Maple [88] and Mathematica [227] can be of great help.

For linear models the *Laplace transform* or *transfer function* approach can be applied. For nonlinear models the oldest method and most simple to understand is the *Taylor* or *power series expansion* [209]. The observable function is expanded in a Taylor series at a particular time point. The time derivatives are evaluated in terms of the parameters resulting in a system of nonlinear equations for the parameters. If this system has a unique solution the model is structurally identifiable. For simple examples using the Laplace transform (linear model) and Taylor series (Michaelis-Menten kinetics) we refer to Godfrey and Fitch [95]. Another classic method is the *similarity transformation* approach [276] (see also [71,205]). In [41] these two methods have been compared without a decisive preference. Recently methods are developed that use differential algebra techniques (see [8] and references therein).

2.3.1.2 a posteriori identifiability

Although necessary, a priori structural identifiability is obviously not sufficient to guarantee successful parameter estimation from real data, and this is when the concept of a posteriori or practical identifiability comes into play. One still assumes that the model structure is exact, however, now data are sparse and noisy and the question is: can the unknown parameters of the postulated model be uniquely estimated from the (possibly noisy and scarce) available data?

A posteriori Identifiability is the logical step that follows the model fitting. A priori identifiability guarantee the existence of not of a solution, but it does not guarantee successful parameter estimation from real data.The difficulty in estimating the parameters in a quantitative mathematical model is not so much how to compute them, but more how to assess the quality of the obtained parameters, since this does not only depend on how well the model describes the phenomenon studied and on the existence of a unique set of parameters, but also on whether the experimental data are *sufficient in number*, *sufficiently significant* and *sufficiently accurate*. With respect to the first two requirements, a sufficient and significant amount of data, it is clear that, whatever method one uses to fit a model with experimental data: to estimate m unknown parameters one needs at least m experimental values. On the other hand, it is not necessary to have experimental data for all state variables involved in the model at all possible time points, often only a few measurements for the *right* observable at *significant* times are needed.The last question, sufficiently accurate data, is related to the fact that measurement errors imply that we do not

have precise data points to fit our model with, but that each point represents a whole cloud of possible data values, implying that also the inferred parameters are not point-values but are contained in a cloud. Depending on the model the cloud of possible parameter values varies in size and shape and can be much larger than the original uncertainty in the data. Summarising, the questions addressed by the a posteriori identifiability are:

- How good is the precision of the parameter estimates?
- Can the model parameters be estimated from the data with acceptable statistical precision?
- Does the model predict the data in an acceptable way? The goodness-of-fit must be acceptable
- How good is the precision of the parameter estimates?

2.3.2 Sensitivity/Uncertainty analysis

Once parameter estimates are obtained by means of optimization, it is important to address the precision of those parameters, since in general, the optimization yields large parameter uncertainties [124, 175]. Analyzing the effects of parameter uncertainty on results is a primordial step in modelling, especially for high dimensional complex models. Beside learning the level of confidence in parameter estimates, uncertainty analysis is useful to discriminate between competitive models. It provides insight into the various source of uncertainty that are important or not with respect to a given response. Different methods can be employed to address the uncertainty analysis of parameter estimates such as sensitivity testing, analytical methods, sampling based methods and computer algebra based methods. Sensitivity testing aims at studying model response or robustness with respect to change within the model structural formation or parameter variation.
The confidence interval (CI) gives a realistic measure of the precision of the parameters. Fundamental for making predictions is the assessment of uncertainties; if the confidence interval on a prediction is too large, the prediction cannot be used for model validation [40]. Beside linear model for which it exists theoretical exact methods to evaluate confidence intervals, in most cases, approximation methods are used [228].

2.4 Discussion

The aim of this chapter was to give a comprehensive survey of parameter estimation, i.e., to discuss both the methods to fit the parameters of a mathematical model to experimental data and to analyze the results. A recent review paper of van Riel [279] discusses these subjects more from the perspective of systems biology but less extensively.

An optimal use of the methods, especially of the global ones, is problem - dependent and, in practice, convergence to the minimum is not guaranteed. Global methods are often used with a computational time limit toprevent an endless search and local methods can get stuck in a local minimum. In general, a good initial guess e.g. from experiments will not be available for all parameters, ruling out the option of using only local search methods. A good strategy is often to use global search methods to find various "promising" areas in the parameter space. Once in these areas, local search methods converge much faster to the minimum (see, e.g. [6,78,228,281] and Section 2.2.4). Since global methods explore the complete "fitness landscape" it is also possible to find multiple parameter vectors that satisfy the experimental data.

In Section 2.2.2 we compared several algorithms for global search. Simulated Annealing and Branch and Bound have a proper convergence theory. The disadvantage of B&B is that it can only be applied if it is possible to compute lower and upper bounds for the objective function. SA is generally applicable, but the theoretical convergence is in practice not much worth since it is critically dependent on the cooling-down schedule. At each temperature the inner-loop (Metropolis) needs to be iterated long enough to explore the regions of search space. However, the balance between the maximum step size and the number of Monte Carlo steps is often difficult to achieve, and depends very much on the characteristics of the search space or energy landscape. SA is computationally very expensive and is not easily paralellizable.

Evolutionary algorithms consistently perform well for all types of problems and are well-suited to solve problems with a truly large search space. The critical factor to escape local minima is the cross-over operator that allows each individual to explore other possibilities by means of information transfer [149]. The critical factor for fast convergence is the selection operator. Premature convergence occurs if an individual that is more fit than most of its competitors emerges too early, it may reproduce so abundantly that it drives down the population's diversity too soon. This will lead the algorithm to converge to the local optimum of that specific individual rather than searching the fitness landscape thoroughly enough to find the global optimum [81]. For a proper behavior the population size should be sufficiently large which means that the method is expensive if the computation of the objective function is not extremely cheap. Fortunately, EA is intrinsically parallel. Multiple individuals can explore the search space in different directions. In contrast to SA, EA can be implemented as a self-tuning method, the most successful example is SRES [233]. For most problems an evolutionary algorithm, like SRES, is robust and easy to use.

The local search methods are introduced in Section 2.2.3. Direct-search methods are generally applicable, but they are less efficient especially for high-dimensional problems. If possible, i.e., if the problem is smooth we recommend to use Newton or trusted region and for a least-squares fit Levenberg-

Marquardt. In non-smooth problems the objective function is discontinuous or has a discontinuous derivative, e.g., because the mathematical model contains step-functions, absolute values, if-then-else constructions, etc. In this case gradient-based methods can not be applied. The Hooke-Jeeves method, or more generally, the generating set search methods are reliable but slow. The Nelder-Mead simplex method is in most cases efficient, but it can fail unpredictably [173].

Normally, the methods described here are used as *single shooting* methods, meaning that the integration path leading to the observable function value in the objective function is determined by the initial conditions for the state variables. Especially, if these initial conditions depend on parameters this can lead to the wrong minimum. To avoid this, one can use the *multiple shooting* approach [271] where the time interval is partitioned and new initial conditions are used at the start of each part of the interval. To connect the integration paths smoothly an augmented system has to be solved. Here the optimal method choice is dependent on the objective funtion and on the DAE system. For a least-squares fit and smooth problems we recommend Levenberg-Marquardt. If the (derivative of) the objective funtion is discontinuous a direct method like Nelder-Mead should be used. If the initial conditions of the DAEs depend also on the parameters and the solution of the DAE system depends strongly on the initial conditions, the multiple shooting strategy could be advantageous. A promising, but not yet fully tested strategy is the intrinsic global-local approach implemented in SSKm (cf. Section 2.2.4). Most important: for all optimization algorithms it is the implementation that counts, especially if the parameter space is restricted by constraints (see also Section 2.2.5).

Finally, finding a parameter vector is only half the job. It is important to study how robust against perturbations the parameters are. If the objective function is the Maxium Likelihood estimator (2.4), the analysis method described in Section 2.3.1.2 can be applied. Otherwise one can use a repeated fitting strategy [109] to study the fitness landscape.