Automatic method for coding large systems of ODEs

Sometimes we want to simulate a system with a large number of ODEs that are structurally similar, e.g. many different species but with homogeneous parameters, or a single species in lots of spatial compartments, etc. Hand-coding all these ODEs is laborious and error-prone. Here is a method for doing it automatically. First we will cover the general version then show an example small system in more detail.

General case

Consider a model with N different molecular species each with concentration x_i, where $i \in [1, \ldots, N]$ indexes the species. A subset of P of the pairs from these species can bind. The concentrations of these products are also denoted x_i but with $i \in [N+1, \ldots, N+P]$. So in total we have $D = N + P$ species, with one ODE for each. Let $X = [x_1, x_2, \ldots, x_D]$ be the vector of all the concentrations. In general,

$$\frac{dX}{dt} = f(X, \Theta)$$

where $\Theta = \{k_1, \ldots, k_{total}\}$ is the list of reaction rates.

However in order to make the coding easier we will be rewriting this a matrix equation:

$$\frac{dX}{dt} = MY$$

where $M$is a matrix that contains all the reaction terms. It will have a number k_j at the position corresponding to each reaction, and zeros elsewhere. Y is a vector that is directly related to the state vector X. However instead of just having the states in it, each element corresponds to one term in the reaction system. This may be bigger or smaller than the state vector itself and will usually include some of the following:

- Individual species
- Bimolecular species
- Products of any pairs of individual species that react
- A 1, to handle spontaneous creation of any species

Small example

Take the following system with three types of molecules a, b, and c:

$\emptyset \xrightarrow{k_1} a$

$a \xrightarrow{k_2} \emptyset$

$a + b \underset{k_4}{\overset{k_3}{\rightleftharpoons}} ab$

$b + c \underset{k_6}{\overset{k_5}{\rightleftharpoons}} bc$

The total system has five species: $X = [a, b, c, ab, bc]^T$. Their corresponding ODEs are

$\frac{da}{dt} = k_1 - k_2 a - k_3 a \times b + k_4 ab$

$\frac{db}{dt} = -k_3 a \times b + k_4 ab - k_5 b \times c + k_6 bc$

$\frac{dc}{dt} = -k_5 b \times c + k_6 bc$

$\frac{dab}{dt} = k_3 a \times b - k_4 ab$

$\frac{dbc}{dt} = k_5 b \times c - k_6 bc$

Now the next step is write down the vector Y which contains all the terms in the system of ODEs. For the above system it has six elements:

$$Y = [a, ab, bc, a \times b, b \times c, 1]$$

Finally, we can write the reaction matrix. It should have as many rows as the number of species/ODEs in the system (5 in our case), and as many columns as the number of elements in Y (6 in our case).

$$M = \begin{bmatrix} -k_2 & k_4 & 0 & -k_3 & 0 & k_1 \\ 0 & k_4 & k_6 & -k_3 & -k_5 & 0 \\ 0 & 0 & k_6 & 0 & -k_5 & 0 \\ 0 & -k_4 & 0 & k_3 & 0 & 0 \\ 0 & 0 & -k_6 & 0 & k_5 & 0 \end{bmatrix}$$

Coding

To code this I suggest writing a function `calcDerivative(X,M)` which takes as input the current state X and the reaction matrix M, and returns the vector of derivatives $[\frac{dx_1}{dt}, \frac{dx_2}{dt}, \ldots \frac{dx_D}{dt}]$. This is passed to the ODE solver.

This function will need to call another function inside it to calculate Y given any particular X. And before running the simulation you will precompute the reaction matrix M. This can usually be done by via a series of for loops, one for each type of reaction. The idea is that each loop iterates through all the elements of the matrix and inserts the reaction rate values in the correct places. If there are some identical reactions that are repeated over and over again, the corresponding matrix elements can be inserted en masse.