Computing Steady States with Stan's Nonlinear Algebraic Solver

Charles Margossian

Columbia University, Department of Statistics
(formally Metrum Research Group LLC)

January 11th 2018

Root-finding

Given a function $f(x, ...)$,

find x^* such that $f(x^*, ...) = 0$.

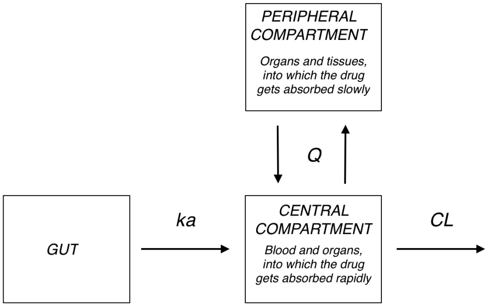
Solving $f(x^*, ...) = 0$ has applications in:

- physics
- astronomy
- biomedicine
- econometrics
- ecology
- and more.

Solving $f(x^*, ...) = 0$ has applications in:

- physics
- astronomy
- biomedicine: characterizing patients at steady states.
- econometrics
- ecology
- and more.

- Drug diffusion in the body:



- ODEs describing the drug diffusion model:

$$\frac{dy_{\mathrm{gut}}}{dt} = -k_a y_{\mathrm{gut}}$$

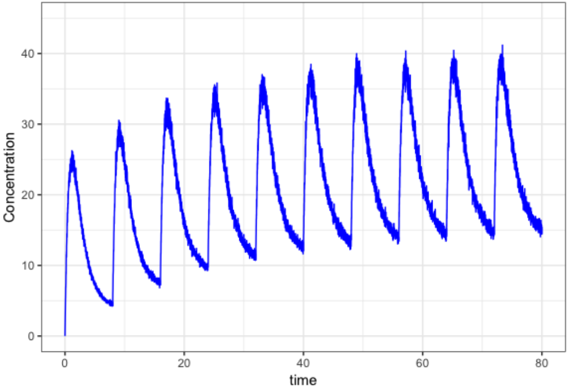
$$\frac{dy_{\mathrm{cent}}}{dt} = k_a y_{\mathrm{gut}} - \left(\frac{CL}{V_{\mathrm{cent}}} + \frac{Q}{V_{\mathrm{cent}}}\right) y_{\mathrm{cent}} + \frac{Q}{V_{\mathrm{peri}}} y_{\mathrm{peri}}$$

$$\frac{dy_{\mathrm{peri}}}{dt} = \frac{Q}{V_{\mathrm{cent}}} y_{\mathrm{cent}} - \frac{Q}{V_{\mathrm{peri}}} y_{\mathrm{peri}}$$

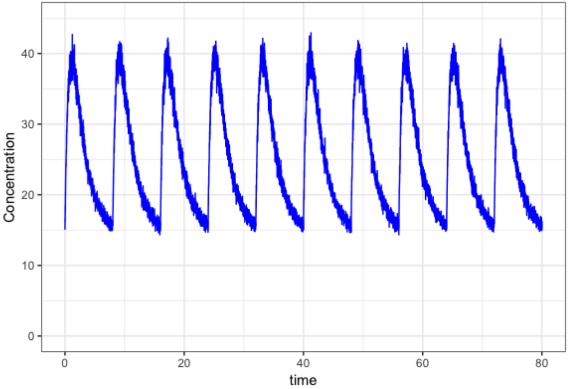
Treatment Cycle

Time	Event
t_0^-	Beginning of the cycle.
t_0^+	Patient receives a dose.
t_0^+ to $t_0 + \tau$	Drug distributes in the body and partially gets cleared.
$t_0 + \tau$	Next cycle begins.

Concentration at the beginning of the treatment



Concentration at steady state



- Let $y(t)$ be the drug mass vector in the patient's body (i.e. in all compartments of the model).
- Let τ be the interdose-interval.
- Steady state is reached when:

$$\mathbf{y}(t + \tau) = \mathbf{y}(t)$$

$$\mathbf{y}(t+\tau) = \mathbf{y}(t)$$
$$\iff \mathbf{y}(t+\tau) - \mathbf{y}(t) = 0$$

The drug mass **y** depends on:

- y_0 (initial drug mass)
- t: time
- θ: the model parameters
- x: fixed data

Need to find y^* such that:

$$\mathbf{f}(y^*, t, ...) = \mathbf{y}(y^*, t, ...) - \mathbf{y}(y^*, t + \tau, ...) = 0$$

Augmented root-finding problem

For Hamilton Monte Carlo sampling [1, 2, 3] need:

- y^*
- J: the Jacobian of the solution with respect to the parameters θ.

$$J^* = \begin{bmatrix} \frac{\partial y_1^*}{\partial \theta_1} & \dots & \frac{\partial y_1^*}{\partial \theta_n} \\ \dots & \dots & \dots \\ \frac{\partial y_n^*}{\partial \theta_1} & \dots & \frac{\partial y_n^*}{\partial \theta_n} \end{bmatrix}$$

A glimpse at the algorithms I

- y^* is computed using a modification of Powell's Hybrid method [4], as implemented in Eigen [5], which is itself based on the MINPACK-1 implementation [6].
- Three tuning parameters:
  - relative tolerance
  - max number of steps
  - Function tolerance: how close is $||f(y^*)||$ to 0?

A glimpse at the algorithms II

- The sensitivities is obtained using a lemma of the implicit function theorem:

$$J^* = -[J_y]^{-1} J_\theta$$

.

- Need J_y to be invertible.
    - Hence the number of unknowns and equations must be the same.

Coding the algebraic equation

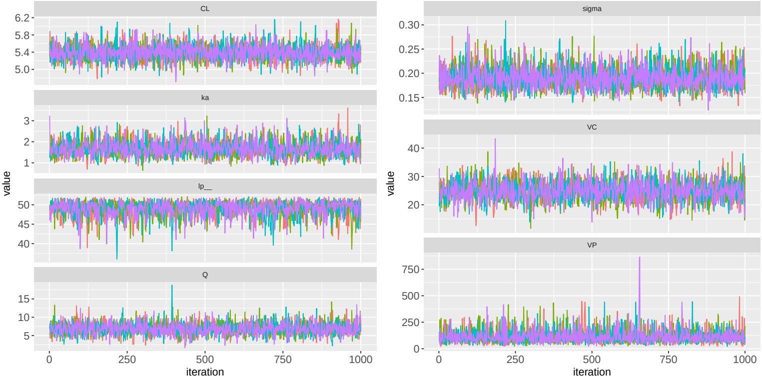
```
vector system(vector y, // unknown
              vector theta, // vector of parameters
              real[] x_r, // real data
              real[] x_i) // integer data
{
  vector[2] z;
  z[1] = y[1] - theta[1];
  z[2] = y[1] * y[2] - theta[2];
  return z;
}
```

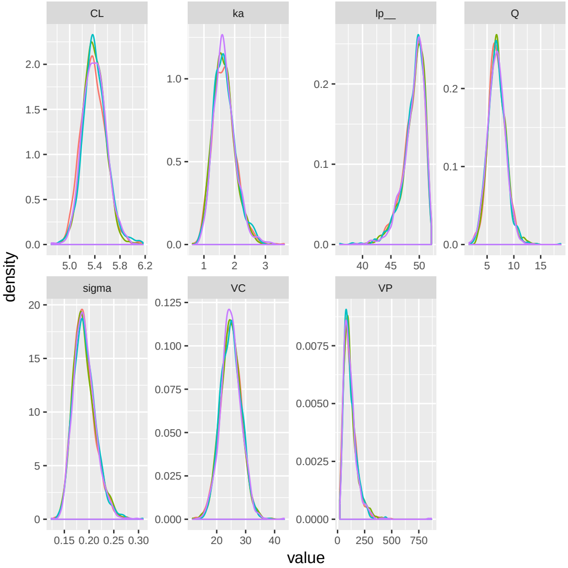
Code for the algebraic equation II

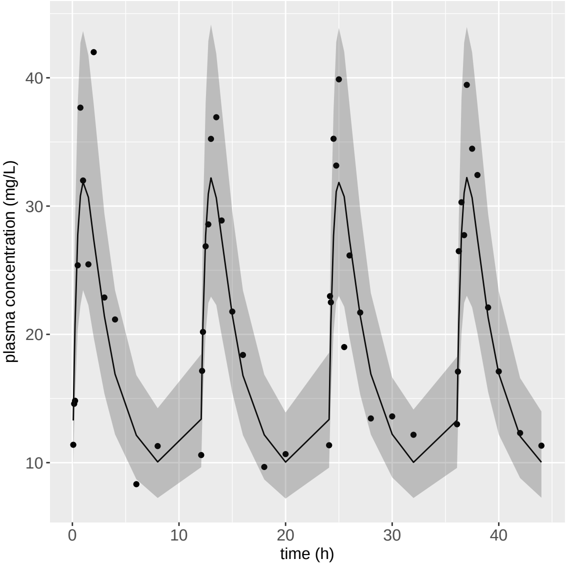
```
vector f(vector y, vector theta, real[] x_r, int[] x_i)
{
  real amt = x_r[2];
  int cmt = x_i[1];
  real y_ii[3] = to_array_1d(y);
  y_ii[cmt] = y_ii[cmt] + amt;
  y_ii = integrate_ode_rk45(twoCptModelODE, y_ii, 0,
                            rep_array(x_r[1], 1),
                            to_array_1d(theta),
                            rep_array(0.0, 1),
                            rep_array(0, 1))[1];

 // return difference between evolved and initial state
 return to_vector(y_ii) - y;
}
```

Example code for algebraic solver

```
y0 = algebra_solver(system, // algebraic equation
                    y_init,
                    theta, x_r, x_i,
                    rel_tol, f_tol, max_num_steps) ;
```


CL
6.2
5.8
5.4
5.0
ka
3
2
1
lp__
50
45
40
Q
15
10
5
value
0
250
500
750
1000
iteration
sigma
0.30
0.25
0.20
0.15
VC
40
30
20
VP
750
500
250
0
value
0
250
500
750
1000
iteration


CL
ka
lp__
Q
sigma
VC
VP
density
value
2.0
1.5
1.0
0.5
0.0
5.0
5.4
5.8
6.2
1.0
0.5
0.0
1
2
3
0.2
0.1
0.0
40
45
50
0.2
0.1
0.0
5
10
15
20
15
10
5
0
0.15
0.20
0.25
0.30
0.125
0.100
0.075
0.050
0.025
0.000
20
30
40
0.0075
0.0050
0.0025
0.0000
0
250
500
750


plasma concentration (mg/L)
10
20
30
40
0
10
20
30
40
time (h)

Acknowledgment

- Individuals:
  - Bill Gillespie (Metrum Research Group)
  - Bob Carpenter (Columbia)
  - Michael Betancourt (Columbia)
  - Ben Goodrich (Columbia)
- Institutions:
  - Office of Naval Research, Bill & Melinda Gates Foundation
  - Columbia University, Metrum Research Group, AstraZeneca

References I

[1] Radford M. Neal.
MCMC using Hamiltonian Dynamics.
Handbook of Markov Chain Monte Carlo. Chapman & Hall / CRC Press, 2010.

[2] Michael Betancourt.
A conceptual introduction to hamiltonian monte carlo.
arXiv:1701.02434v1, January 2017.

[3] Matthew D. Hoffman and Andrew Gelman.
The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo.
Journal of Machine Learning Research, pages 1593–1623, April 2014.

[4] Michael J. D. Powell.
A hybrid method for nonlinear equations.
In P. Rabinowitz, editor, *Numerical Methods for Nonlinear Algebraic Equations*. Gordon and Breach, 1970.

[5] Gaël Guennebaud, Benoît Jacob, et al.
Eigen v3.
`http://eigen.tuxfamily.org`, 2010.

References II

[6] Kenneth E. Hillstrom Jorge J. More, Burton S. Garbow.
User Guide for MINPACK-1.
Argonne National Laboratory, 9700 South Cass Avenue, Argonne, Illinois 60439, August 1980.