A directly Programmed Solution to ARGESIM Comparison C15 'Clearance Identification" with Java and JMSL Numerical Library

Daniel Leitner, F. Breitenecker, ,Vienna Univ. of Technology; dleitner@osiris.tuwien.ac.at

Simulator: Java 5 is an object oriented platform independent programming language. JMSL is a numeric library for Java which is little object oriented but favours static methods. JMSL is a successor of IMSL which is a numerical library for FORTRAN and is written 100% in Java.

Model: The model was implemented in an object oriented manner. The class `CompartmentModel` represents the model, the class `Compartment` a compartment and the class `Connection` the flux from one Compartment into another.

```
CompartmentModel model=new CompartmentModel();
Compartment Vc=new Compartment("central",0);
Compartment Vp=new Compartment("peripheral",0);
model.add(Vc);
model.add(Vp);
Connection k1i=new Connection(null,Vc);
Connection ko1=new Connection(Vc,null,0.0041);
Connection k21=new Connection(Vc,Vp,0.0498);
Connection k12=new Connection(Vp,Vc,0.0585);
model.setVolume(7.1);
```

This allows extracting the ODE model with `model.getLinearODE()` and can be solved by the static methods of the JMSL library.

Task a: Simulation of the System. The ODEs are solved by the `OdeRungeKutta` class of JMSL. It implements Runge-Kutta-Verner fifth-order and sixth-order method. The plots where create with JMSL chart package

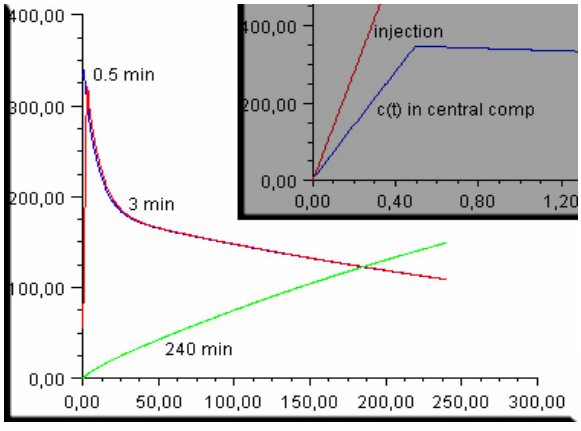

Fig. 1: Concentrations for different injections

Different injection times result in different concentrations in the central compartment (Fig. 1). The difference between the concentration of the injection and the real concentration in the central compartment can investigated in the grey plot (Fig. 1). After 0.5 min the injection stops and the concentration starts to decrease. The calculated values one minute after injection for t1=0.5, t2=3 and t3=240 are 2342.46 (t=1.5), 2208.84 (t=4) and 1060.4 (t=240).

Task b: Parameter Identification. The identification is done with a modified Levenberg-Marquardt method which is implemented in JMSL. The class is called `BoundedLeastSquares`. To make sure the numerical function is evaluated at exactly the same points the experimental data is measured, the Runge-Kutta solver has to integrate between the given times.

```
for (int i=0; i<times.length-1; i++) {
  solver.solve(times[i],times[i+1],y);
  f[i+1]=fehler(y);
}
```

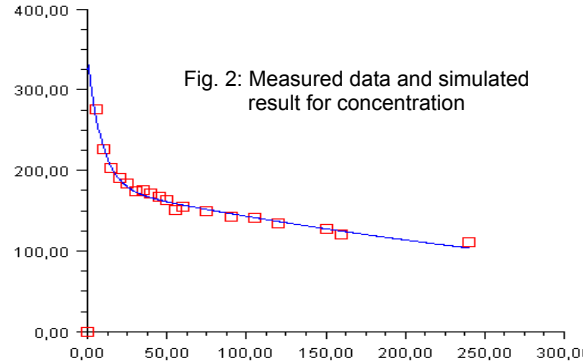

Fig. 2: Measured data and simulated result for concentration

Results of the identification are shown in Figure 2, the identified parameters are given in the next table:

k_{01}	k_{21}	k_{12}	V_1	Max	Clear.	Res
.0043	.0501	.0599	7.28	330	31.34	288

Task c - Error Estimation. The data are disturbed using `Random.nextNormalAR()` from JMSL, an acceptance/rejection algorithm to compute standard normal distribution. Mean and standard deviation of 1000 samples are given in the next table:

	k_{01}	k_{21}	k_{12}	V_1
mean	.0042	.0506	.0595	7.297
sd	2.1E-7	.000105	.000057	.1484

C15 Classification: Programmed Numerical Appr.

Simulator: Java 5, JIMSL