Classroom Tips and Techniques: Estimating Parameters in Differential Equations

Robert J. Lopez
Emeritus Professor of Mathematics and Maple Fellow
Maplesoft

Initializations

```
> restart;
> interface(warnlevel=0):
  with(plots):
  with(Optimization):
  with(LinearAlgebra):
  with(Statistics):
  with(DEtools):
>
```

Introduction

In the previous two articles in this series, we showed several methods for estimating parameters in a mathematical model. For example, we discussed the problem of fitting a circle to three-dimensional data, and fitting an ellipse to data in the plane. In this article, we consider estimating the parameters in a differential equation that governs a physical system from which we've extracted observational data. Although our technique would work for a boundary value problem, we will restrict our discussion to initial value problems.

If the initial value problem has an exact solution, the parameters in its differential equation may be estimated by a (possibly nonlinear) least-squares fitting technique applied to the solution. However, if a closed-form solution is not available, it is still possible to apply a numeric least-squares process reminiscent of the shooting method used to solve boundary value problems.

For each point P in the space of parameters for the differential equation, the initial value problem can be integrated numerically and a vector of values for its solution computed and compared to a vector of observed values. The sum of squares of deviations would be the L_2-norm of the difference of these vectors,

and minimizing this norm should provide a least-squares fit of the solution function to the data.

Our two examples, one for a single equation, and one for a system, will illustrate how to add random noise to observations and how to write and minimize a function defined by the numeric solution of differential equations.

Example 1: Parameter Estimation with Noisy Data

The Model

Consider the nonlinear damped oscillator in which the damping force is assumed proportional to the velocity, a system governed by the nonlinear differential equation

```
> DE := diff(y(t),t,t) + a*diff(y(t),t)*abs(diff(y(t),t)) + b*y(t)
  = 0;
```

$$DE := \frac{d^2}{dt^2} y(t) + a \left(\frac{d}{dt} y(t) \right) \left| \frac{d}{dt} y(t) \right| + b y(t) = 0 \qquad \textbf{(3.1.1)}$$

```
>
```

The unknown parameters are the damping coefficient a and the spring constant b.

To simulate observational data, we start with the model

$$y'' + 2 y' |y'| + 4 y = 0$$

$$y(0) = \frac{1}{2}$$
$$y'(0) = 2$$

compute equi-spaced displacement $y(t)$, then add random noise to the observations.

Generating the Observations

Starting with the differential equation

```
> q := eval(DE, [a=2, b=4]);
```

$$q := \frac{d^2}{dt^2} y(t) + 2 \left(\frac{d}{dt} y(t) \right) \left| \frac{d}{dt} y(t) \right| + 4 y(t) = 0 \qquad \textbf{(3.2.1)}$$

```
>
```

and the initial conditions

```
> ic := y(0)=1/2, D(y)(0)=2;
```

$$ic := y(0) = \frac{1}{2}, \mathrm{D}(y)(0) = 2 \qquad \textbf{(3.2.2)}$$

```
>
```

we obtain a numeric solution via

```
> Q := dsolve({q,ic}, y(t), numeric, output=listprocedure):
>
```

The function

```
> Yt := rhs(Q[2]):
>
```

is a Maple-generated procedure that provides $y(t)$ numerically. Choosing

```
> N := 50;
  d := .2;
```

$$N := 50$$
$$d := 0.2 \qquad \textbf{(3.2.3)}$$

```
>
```

as the number of observations and the time interval between observations, we write

```
> Y := [seq(Yt(d*k),k=0..N)]:
>
```

to obtain a list of $N + 1$ observations, and

```
> P := [seq([d*(k-1),Y[k]], k=1..N+1)]:
>
```

to obtain a list of corresponding points.

Figure 1 contains a graph of the numeric solution of the governing initial value problem, with the computed points superimposed.

```
> p1 := odeplot(Q,[t,y(t)],0..10, numpoints=100, color=red):
  p2 := plot(P, style=point, color=black):
  display([p1,p2]);
```

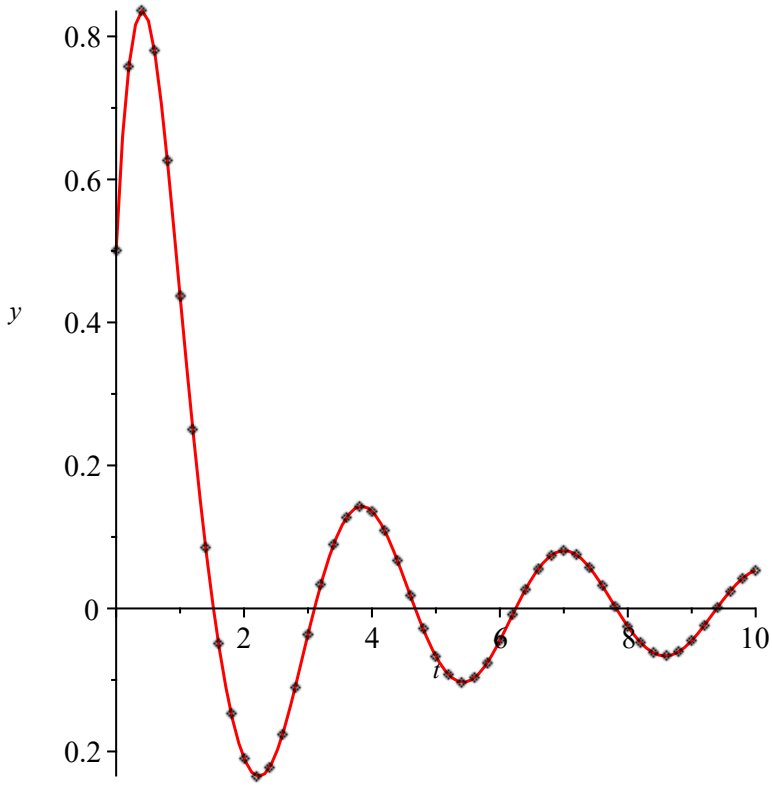

Figure 1 Numeric solution of governing initial value problem

```
>
```

Adding Random Noise

We distinguish between two types of random noise appearing in experimental data. True experimental noise is relative to the data-collection process itself. External interference is additive.

A sample of size $N + 1$ taken from a normal distribution with mean 1 and standard deviation σ provides factors normally distributed about 1. Multiplying each computed value of $y(t)$ by one such factor perturbs the computed values with noise of magnitude relative to that of the computed value.

Choosing $\sigma = 0.3$, we have

```
> S := Sample(Normal(1,.3),N+1):
  Yn := zip(`*`, S, Vector(Y))^%T:
>
```

for our noisy data. Figure 2 shows the noisy observations as red dots, and the original computed displacements as black dots.

```
> p3 := plot([seq([d*(k-1),Yn[k]],k=1..N+1)],style=point, color=
  red):
  display([p2,p3]);
```

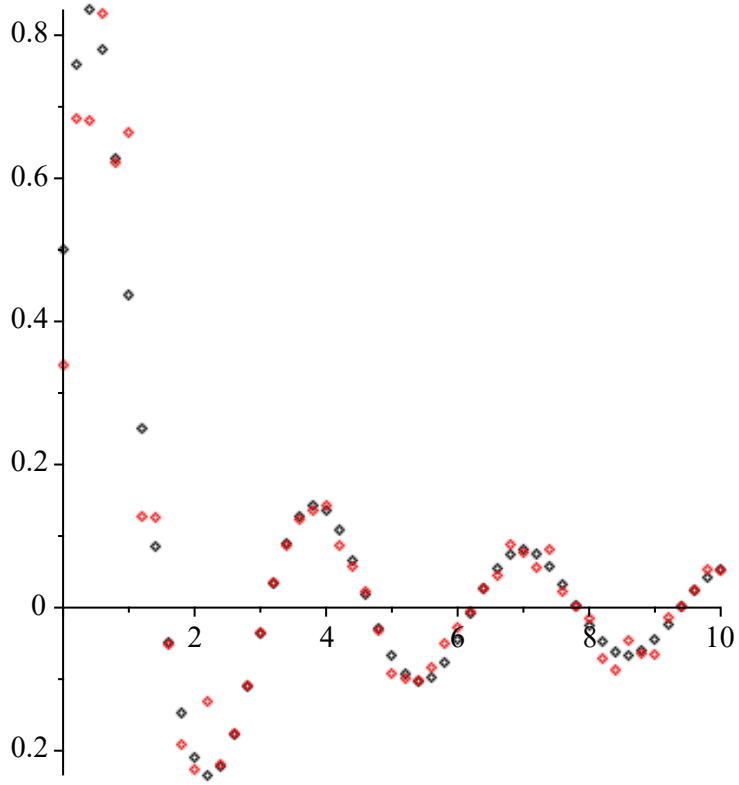

Figure 2 Computed displacements (black) and noisy observations (red)

The Sum of Squares of Deviations

The following Maple procedure is a function of the two parameters a and b appearing in the differential equation of our nonlinear model. For each pair (a, b), the differential equation is integrated numerically so that $N + 1$ equispaced nodes are computed. These nodes are compared to the noisy observations via the Euclidean norm of the difference between the vector of observations and the vector of computed solution values.

```
> SS := proc(a,b)
  local F, V;
  if not type([a,b],[numeric,numeric]) then return 'SS'(a,b);
  elif a<0 or b<0 then return 1e100;
  end if;
  F := dsolve(eval({DE,ic},{:-a=a,:-b=b}), y(t), numeric, output=
  Array([seq(d*k,k=0..N)]));
  V := convert(Column(F[2,1],2),Vector);
  Norm(V-Yn,2);
  end proc:
```

(The penalty-function construct that keeps the parameters in the first quadrant was suggested by Dr. Allan Wittkopf of Maplesoft.)

Figure 3 contains a graph of the surface $SS(a, b)$, which indicates that there is indeed at least one minimum for this function.

```
> plot3d(SS,0..5, 2..6, axes=box, view=.3..3/2,
      labels=[a,b,""], orientation=[-65,70]);
```

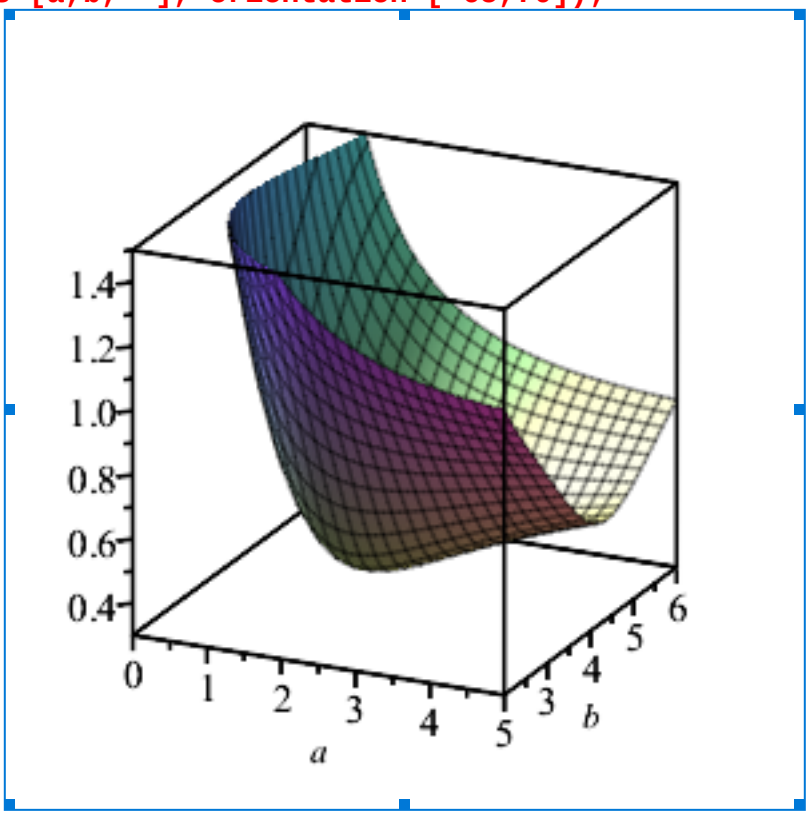

Figure 3 The surface $SS(a, b)$ showing existence of a minimum

>

Minimization

What remains is to compute a minimum, a task Maple performs with its **NLPSolve** command from the *Optimization* package. By option, this command can implement the Nelder-Meade (simplex) algorithm that is Maple's only direct method - it does not differentiate the objective function. In two dimensions, the simplex is a triangle. Function values are obtained at the vertices of the triangle, and for a minimization, the triangle is reflected away from the vertex with the highest value. Various scaling and anti-cycling devices are included in the algorithm.

Using this command, we determine

```
> R := NLPSolve(SS(a,b), method=nonlinearsimplex, initialpoint=[a=
  5,b=5], evaluationlimit=250);
```

$$R := [0.380704346312640995, [a = 2.03632981631742, b = 4.00480923066649]] \quad \textbf{(3.5.1)}$$

```
>
```

where the first number in the output list is the minimum function value, and the sublist contains the minimizing values of the parameters a and b. Surprisingly, they are very close to $(2, 4)$ used to generate the noisy obserations.

The optimum values of the parameters, namely

```
> A := eval(a,R[2]);
  B := eval(b,R[2]);
```

$$A := 2.03632981631742$$
$$B := 4.00480923066649 \quad \textbf{(3.5.2)}$$

```
>
```

cause the differential equation governing our model to become

```
> qq := eval(DE, [a=A, b=B]);
```

$$qq := \frac{d^2}{dt^2} y(t) + 2.03632981631742 \left(\frac{d}{dt} y(t) \right) \left| \frac{d}{dt} y(t) \right| + 4.00480923066649\, y(t) = 0 \quad \textbf{(3.5.3)}$$

```
>
```

A numeric integration of this equation results in Figure 4 where the noisy observations, and two numeric solutions of the model can be seen. In green, we have the solution for the derived parameters, and in red, the solution for the original parameters, $(2, 4)$.

```
> Q := dsolve({qq, ic},y(t),numeric):
  p4 := odeplot(Q,[t,y(t)],0..10, numpoints=100,
                color=green, thickness=3):
  display([p4,p1,p3]);
```

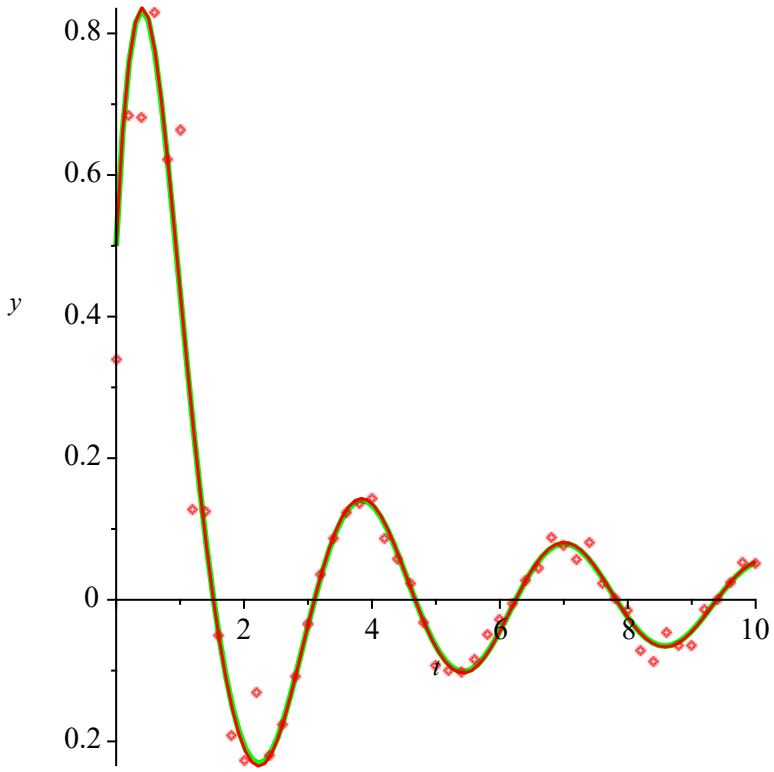

Figure 4 The noisy observations, best-fit solution (green) and unperturbed solution (red)

Example 2: Determining Chemical Rate Constants

In a recent on-line forum, estimating chemical rate-reaction constants a and b for the system governed by the initial value problem

```
> q1 := diff(x(t),t) = -a*x(t)*y(t) + b*z(t);
  q2 := diff(y(t),t) = -a*x(t)*y(t) + b*z(t);
  q3 := diff(z(t),t) =  a*x(t)*y(t) - b*z(t);
  ic := x(0)=10,y(0)=10,z(0)=0;
```

$$q1 := \frac{d}{dt}x(t) = -a\,x(t)\,y(t) + b\,z(t)$$

$$q2 := \frac{d}{dt}y(t) = -a\,x(t)\,y(t) + b\,z(t)$$

$$q3 := \frac{d}{dt}z(t) = a\,x(t)\,y(t) - b\,z(t)$$

$$ic := x(0) = 10, y(0) = 10, z(0) = 0 \qquad \textbf{(4.1)}$$

was proposed. As observed data, the 26 equispaced measurements of $z(t)$, captured in the vector

```
> C := <0,4.981119825867072,6.619863704639184,7.424230767960948,
  7.895291258366989,8.200043893374819,8.410083696006469,
  8.561234239001111,8.673418582102649,8.75860469142499,
  8.824418203860477,8.875947188583103,8.916714710357871,
  8.949234971834077,8.975347266499856,8.996425002439704,
  9.01351139331151,9.027409882146703,9.038747104014094,
  9.048016122443334,9.055608432249834,9.061836841504377,
  9.066952814916256,9.071159284138764,9.074620916958418,
  9.07747159947345>:
>
```

were provided. Figure 5 displays these observations as black circles.

```
> N := Dimension(C)-1:
  Pts := [seq([k-1,C[k]],k=1..N+1)]:
  p5 := plot(Pts, style=point, symbol=circle, color=black, labels=
  [t,"z"]):
  p5;
```

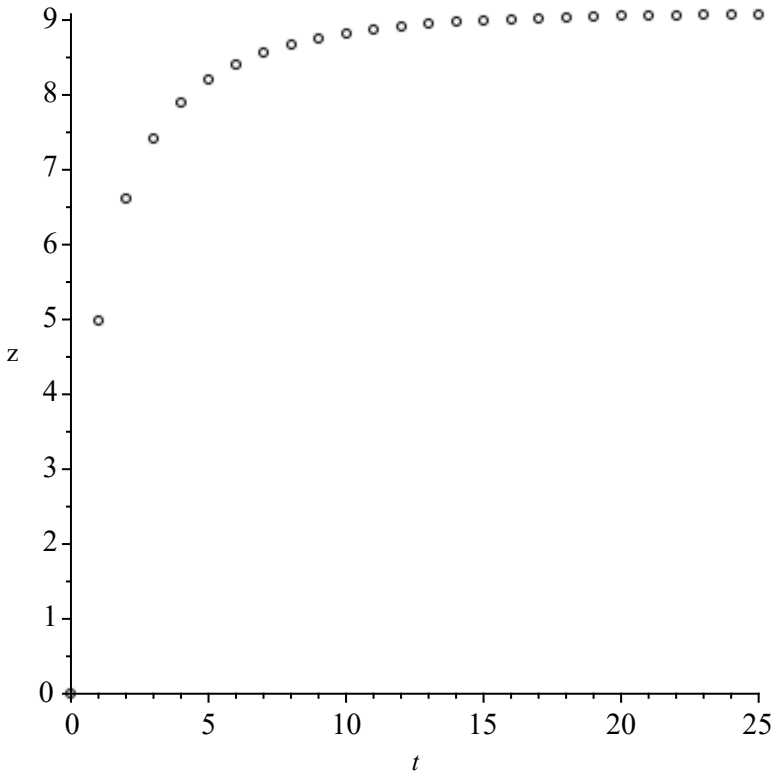

Figure 5 Equispaced observations of $z(t)$

```
>
```

Dr. Allan Wittkopf of Maplesoft showed that because of implied conservation laws, the given system can be reduced to the initial value problem

```
> DE1 := diff(z(t),t)=a*(10-z(t))^2-b*z(t);
  IC := z(0) = 0;
```

$$DE1 := \frac{d}{dt} z(t) = a\,(10 - z(t))^2 - b\,z(t)$$

$$IC := z(0) = 0 \qquad \textbf{(4.2)}$$

```
>
```

In fact, he showed that, in general, the conservation laws

```
> C1 := x(t)+z(t)=x[0]+z[0];
  C2 := y(t)+z(t)=y[0]+z[0];
```

$$C1 := x(t) + z(t) = x_0 + z_0$$

$$C2 := y(t) + z(t) = y_0 + z_0 \qquad \textbf{(4.3)}$$

```
>
```

lead to the single equation

```
> temp := rifsimp([q1,q2,q3,C1,C2],[x,y,z]):
  map(s -> collect(s,{a,b},factor), temp[Solved][1]);
```

$$\frac{d}{dt} z(t) = \left(z(t) - z_0 - y_0\right)\left(z(t) - z_0 - x_0\right) a - b\,z(t) \qquad \textbf{(4.4)}$$

```
>
```

Hence, the exact solution for $z(t)$ is

```
> Zt := rhs(dsolve({DE1,IC},z(t)));
```

$$Zt := -\frac{\tanh\left(\frac{t\sqrt{b\,(40\,a+b)}}{2} + \operatorname{arctanh}\left(\frac{20\,a+b}{\sqrt{b\,(40\,a+b)}}\right)\right)\sqrt{b\,(40\,a+b)} - 20\,a - b}{2\,a} \qquad \textbf{(4.5)}$$

```
>
```

```
> Ztr := simplify(evalc(Zt)) assuming a>0, b>0;
```

$$Ztr := \left(\left(20\,a+b\right)\sinh\left(\frac{t\sqrt{b}\,\sqrt{40\,a+b}}{2} + \frac{\ln\left(20\,a+b+\sqrt{b}\,\sqrt{40\,a+b}\right)}{2} - \frac{\ln\left(20\,a+b-\sqrt{b}\,\sqrt{40\,a+b}\right)}{2}\right) - \cosh\left(\frac{t\sqrt{b}\,\sqrt{40\,a+b}}{2} + \frac{\ln\left(20\,a+b+\sqrt{b}\,\sqrt{40\,a+b}\right)}{2} - \frac{\ln\left(20\,a+b-\sqrt{b}\,\sqrt{40\,a+b}\right)}{2}\right)\right)\sqrt{b}\,\sqrt{40\,a+b}\Big/\left(2\,a\sinh\left(\frac{t\sqrt{b}\,\sqrt{40\,a+b}}{2} + \frac{\ln\left(20\,a+b+\sqrt{b}\,\sqrt{40\,a+b}\right)}{2} - \frac{\ln\left(20\,a+b-\sqrt{b}\,\sqrt{40\,a+b}\right)}{2}\right)\right) \qquad \textbf{(4.6)}$$

```
>
```

This function can be fit to the given observations via the **NonlinearFit** command from the *Statistics* package. In particular, it provides

```
> Params := NonlinearFit(Ztr,<seq(k,k=0..N)>, C, [t], parameternames=
  [a,b], output=parametervalues, parameterranges=[a=0..infinity, b=0..
  infinity]):
  A := eval(a, Params):
  B := eval(b, Params):
  a = A;
  b = B;
```

$$a = 0.100000008713371$$
$$b = 0.00909090959830207 \qquad \textbf{(4.7)}$$

```
>
```

as estimates of the rate constants a and b. Using these values, we obtain Figure 6, a graph of the numeric solution of the original IVP, along with the observed values of $z(t)$.

```
> M := dsolve(eval({q1,q2,q3, ic},[a=.1,b=.00909090090909]),
                    {x(t),y(t),z(t)},numeric):
  p6 := odeplot(M,[t,z(t)],0..25):
  display([p5,p6]);
```

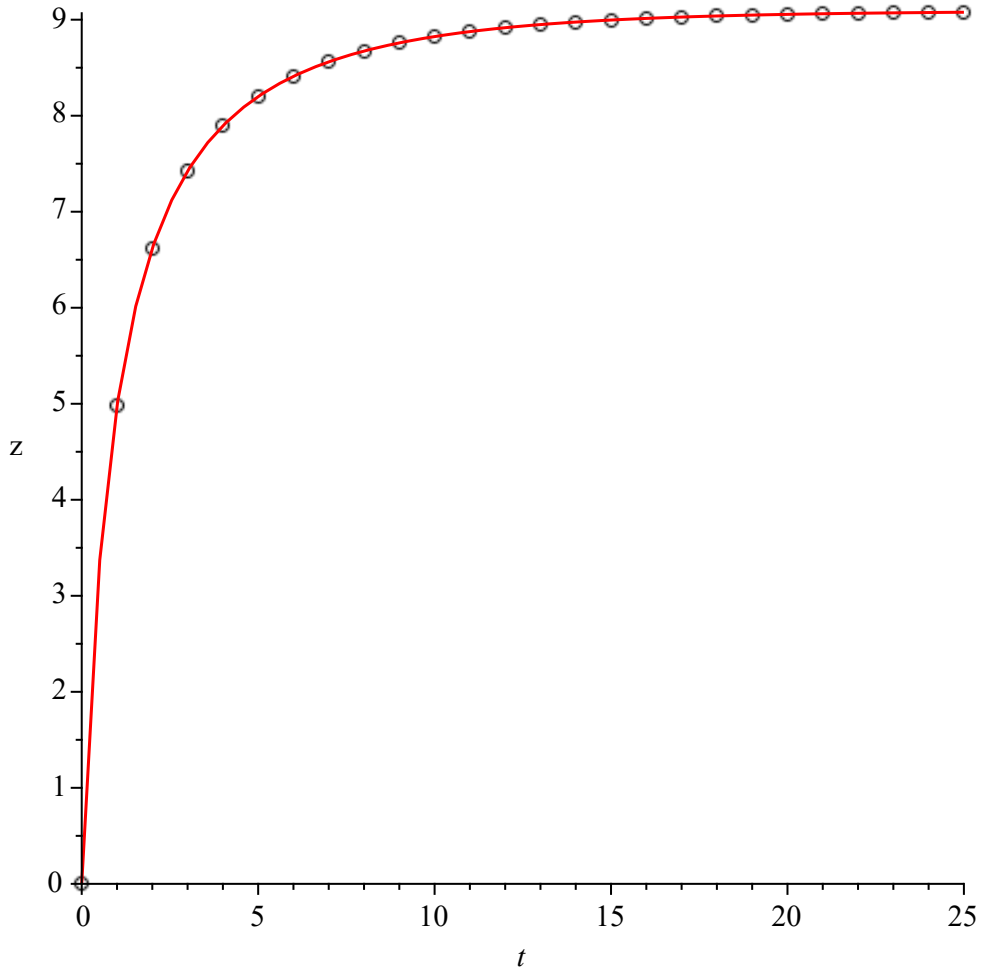

Figure 6 Observations and numeric solution for $z(t)$

>

The parameter-estimation problem for differential equations is an important component of the modeling effort prevalent in the applied sciences. In particular, we point to the extensive work of the control-theory specialist H. T. Banks of North Carolina State University.

Parameters in a system of differential equations that can only be solved numerically can be estimated by the same technique used in Example 1. For each pair of parameteter values (a, b), the function

```
> SS := proc(a,b)
  local F, V;
  if not type([a,b],['numeric','numeric']) then return 'SS'(a,b);
  elif a<0 or b<0 then return 1e100;
  end if;
  F := dsolve(eval({q1,q2,q3,ic},{:-a=a,:-b=b}),[x(t),y(t),z(t)],
  numeric, output=Array([seq(k,k=0..N)]));
  V := convert(Column(F[2,1],4),Vector);
  Norm(V-C,2);
  end proc:
>
```

integrates the given initial value problem numerically, then computes the L_2 norm of the difference between the vectors of computed and observed values of $z(t)$. Hence, $SS(a, b)$, from R^2 to R, represents a sum of squares of deviations.

Seen in Figure 7, the surface determined by $SS(a, b)$ suggests the existence of at least one minimum.

```
> plot3d(SS, .0995..0.1005, .009..0.0092, view=0..0.025,
     axes=box, labels=[a,b,"SS"], orientation=[-80,60]);
```

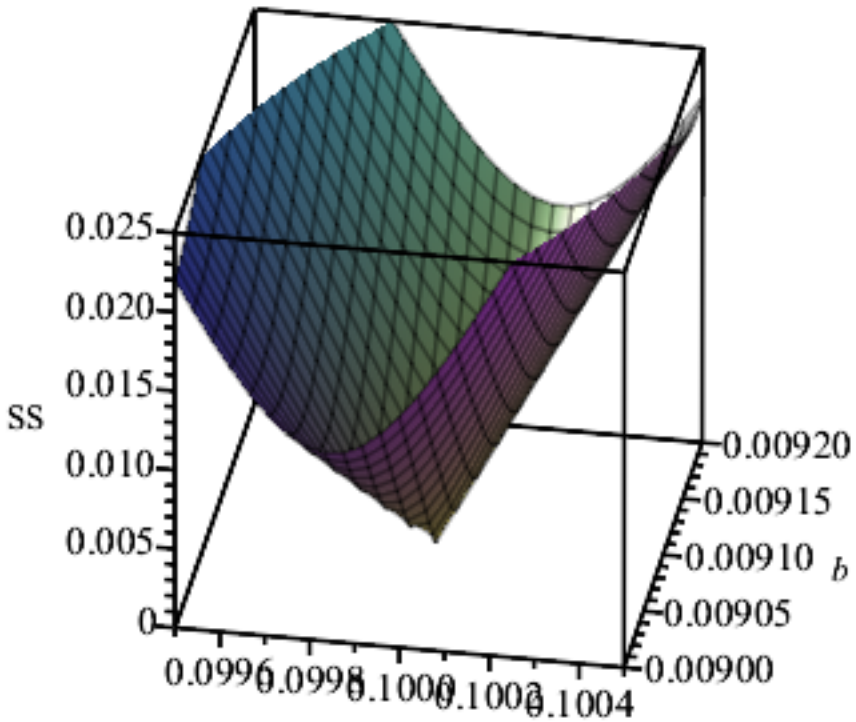

Figure 7 Surface determined by the function $SS(a, b)$

>

To find the minimum shown in Figure 7, we again invoke the Nelder-Meade (simplex) algorithm via the **NLPSolve** command, and obtain

```
> params := NLPSolve(SS(a,b), method=nonlinearsimplex, initialpoint=
  [a=.2, b=.1],evaluationlimit=200):
  An := eval(a, params[2]):
  Bn := eval(b, params[2]):
  a = An;
  b = Bn;
```

$$a = 0.100000023771891294$$
$$b = 0.00909091495408047290 \qquad \textbf{(4.8)}$$

>

as estimates for a and b, with

```
> params[1];
```

$$5.781759478478599921\,10^{-7} \qquad \textbf{(4.9)}$$

>

as the corresponding minimum value of SS. By way of comparison, the values of a and b computed from the exact solution for $z(t)$ differ from these new values by

```
> An-A;
  Bn-B;
```

$$1.51\,10^{-8}$$
$$5.359\,10^{-9} \qquad \textbf{(4.10)}$$

>

respectively, and the minimum of SS at the former point is

```
> SS(A,B);
```

$$9.602838786016251671\,10^{-7} \qquad \textbf{(4.11)}$$

>

a number marginally higher than that determined by our numeric method.

>