Solvers for Stiff Systems

Van der Pol equation

$$z'' - \mu(1 - z^2)z' + z = 0$$

Transform 2nd order ODE
into 2 1st order ODEs:

$$y_1' = y_2$$
$$y_2' = \mu \cdot (1 - {y_1}^2) \cdot y_2 - y_1$$

Initial conditions for the state variables:

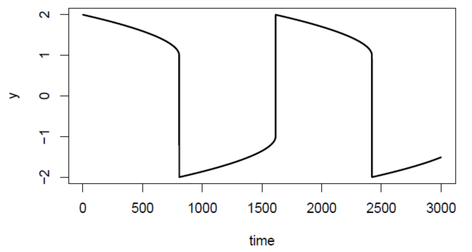
$$y_{1(t=0)} = 2$$
$$y_{2(t=0)} = 0$$

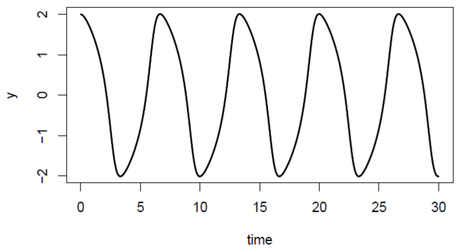
One parameter, μ:
- big value (1000): stiff system
- small value (1) : nonstiff

vanderpol.R

```
library(deSolve)

vdpol <- function (t, y, mu) {
  list(c(
    y[2],
    mu * (1 - y[1]^2) * y[2] - y[1]
  ))
}

yini <- c(y1 = 2, y2 = 0)

stiff <- ode(y = yini, func = vdpol, times = 0:3000, parms = 1000)

nonstiff <- ode(y = yini, func = vdpol, times = seq(0, 30, by = 0.01), parms = 1)


head(stiff, n = 3)

plot(stiff, type = "l", which = "y1",lwd = 2, ylab = "y",main = "IVP ODE, stiff")

plot(nonstiff, type = "l", which = "y1",lwd = 2, ylab = "y",main = "IVP ODE, nonstiff")
```

```
     time       y1            y2
[1,]    0 2.000000  0.0000000000
[2,]    1 1.999333 -0.0006670373
[3,]    2 1.998666 -0.0006674088
```





Try different solver, e.g. "bdf";

```
system.time(
  stiff <- ode(y = yini, func = vdpol, times = 0:3000,
    parms = 1000, method = "bdf")
)

system.time(
  nonstiff <- ode(y = yini, func = vdpol,times = seq(0, 30, by = 0.01),
    parms = 1, method = "bdf")
)
```

Try other solvers:

"ode23", "lsoda", "adams", "bdf", "radau"

more, see ?ode

Result:

solver	non-stiff	stiff
ode23	0.37	271.19
lsoda	0.26	0.23
adams	0.13	616.13
bdf	0.15	0.22
radau	0.53	0.72

Stiff System:

Difficult to give a precise definition.

A system where some components change
much more rapidly than some others.

Difficult to solve:

- solution can be numerically unstable
- may require very small time steps (slow!)
- deSolve contains solvers that are suitable
  for stiff systems

- But: "stiff solvers" less efficient for "well behaving" systems.

solver "lsoda" selects automatically between stiff solver (bdf)
and nonstiff solver (adams)

Solver Overview

Solver	Notes	stiff	y'=f(t,y)	My'=f(t,y)	F(y',t,y)=0	Roots	Events	Lags (DDE)	Nesting
lsoda/lsodar	automatic method selection	auto	x			x	x	x	
lsode	bdf, adams, …	user defined	x			x	x	x	
lsodes	sparse Jacobian	yes	x			x	x	x	
vode	bdf, adams, …	user defined	x				x	x	
zvode	complex numbers	user defined	x				x	x	
daspk	DAE solver	yes	x	x	x		x	x	
radau	DAE; implicit RK	yes	x	x		x	x	x	
rk, rk4, euler	euler, ode23, ode45, … rkMethod	no	x				x		x
iteration	returns state at t+dt	no	x				x		x

- ode, ode.band, ode.1D, ode.2D, ode.3D: top level functions (wrappers)
- red: functionality and/or algorithm added by us

References

Soetaert, K. Petzoldt, T. & Setzer, R. W. (2010):
Solving differential equations in R. The R Journal 2(2), 5-15.