ODE solving Methods

I’m trying to find the maximum active state possible after a given time step. Right now, I’m looking at starting at an active state of 0, giving 1 neural drive and predicting how high the active state can get after the time step (.005 s).

The two methods I’m testing are the ODE45 solution you gave me, and an RK4 method. I want to use rk4 because it is much much faster than ODE45.

I’ll post some sample script here and the results that it gives me.

**ODE45 Solution**

The myode function is the one you sent me. Its pasted below the answer.

time\_step = 0.005;

act\_state = 0;

t\_act = .050;

t\_deact = .066;

sig = 0;

eps = 0;

u\_in = [1,1];

fgt=(0:1)\*time\_step;

f1 = 1./(t\_deact+u\_in\*(t\_act-t\_deact));

g1 = (1+eps\*sig)./(t\_deact+u\_in\*(t\_act-t\_deact));

f2 = 1./(t\_deact);

g2 = (1+eps\*sig)./(t\_deact);

tspan=[0 fgt(end)];

ic = act\_state;

opts = odeset('RelTol',1e-7,'AbsTol',1e-7);

[tnew,anew]=ode45(@(tnew,anew) myode(tnew,anew,fgt,f1,g1,f2,g2,u\_in),tspan,ic,opts);

max = anew(end);

>> max

max =

0.0952

This solution gives a max active state, at 0.0952 given initial active state of 0, with a neural drive of 1.

function dadt=myode(t,a,fgt,f1,g1,f2,g2,u)

u=interp1(fgt,u,t);

f1=interp1(fgt,f1,t);

g1=interp1(fgt,g1,t);

% f2=interp1(fgt,f2,t);

% g2=interp1(fgt,g2,t);

if u>a

dadt=-f1.\*a+g1.\*u;

else

dadt=-f2.\*a+g2.\*u;

end

**RK4 Methods**

One of the methods was taken from <https://www.mathworks.com/matlabcentral/fileexchange/29851-runge-kutta-4th-order-ode?focused=3773771&tab=function>.

The second method using ystar is from <http://lpsa.swarthmore.edu/NumInt/NumIntFourth.html>

t\_act = .050;

time\_step = 0.005;

act\_state = 0;

drive = 1;

t\_deact = .066;

step\_size = .0005;

h = step\_size;

x=0:step\_size:.005;

y = drive\*ones(1,length(x));

y(1) = act\_state;

ystar(1) = act\_state;

F\_y = @(y) (drive-y)/(t\_deact-drive\*(t\_act-t\_deact));

for i=1:(length(x)-1) % calculation loop

k\_1 = F\_y(y(i));

k\_2 = F\_y(y(i)+0.5\*h\*k\_1);

k\_3 = F\_y(y(i)+0.5\*h\*k\_2);

k\_4 = F\_y(y(i)+k\_3\*h);

k1 = (drive-ystar(i))/(t\_deact-drive\*(t\_act-t\_deact)); % Approx for y gives approx for deriv

y1 = ystar(i)+k1\*h/2; % Intermediate value (using k1)

k2 = (drive-y1)/(t\_deact-drive\*(t\_act-t\_deact)); % Approx deriv at intermediate value.

y2 = ystar(i)+k2\*h/2; % Intermediate value (using k2)

k3 = (drive-y2)/(t\_deact-drive\*(t\_act-t\_deact)); % Another approx deriv at intermediate value.

y3 = ystar(i)+k3\*h; % Endpoint value (using k3)

k4 = (drive-y3)/(t\_deact-drive\*(t\_act-t\_deact)); % Approx deriv at endpoint value.

ystar(i+1) = ystar(i) + (k1+2\*k2+2\*k3+k4)\*h/6; % Approx soln

y(i+1) = y(i) + (1/6)\*(k\_1+2\*k\_2+2\*k\_3+k\_4)\*h; % main equation

end

out = y(end);

out = 0.0592

In the rk4 there are actually two solutions, y and ystar. Both are giving me the 0.0592.

