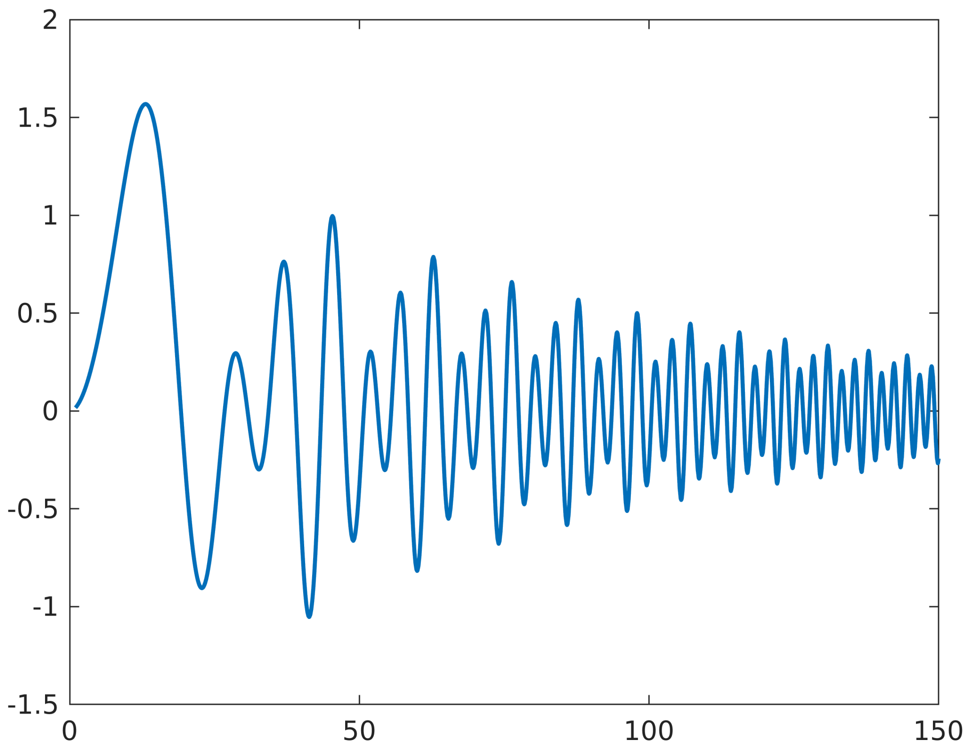
Detecting two components w/ time-varying amplitudes and frequencies

Here we are checking if we can detect the exponential decays and linearly increasing frequencies of a sum of sines.

```
clf
clear

fs =1000;

tspan = 1:0.001:150;
%y_s = 2.*sin(2.*tspan) + sin(0.01.*tspan .* tspan);

y_1 = exp(-0.01.*tspan).*sin(0.01.*tspan.*tspan);
y_1 = y_1';

y_2 = exp(-0.02.*tspan).*sin((1./150).*tspan.* tspan);
y_2 = y_2';



y_s = y_1 + y_2;
tspan = tspan';
```

```
clf
plot(tspan, y_s,'LineWidth', 1.5)
hold on
```



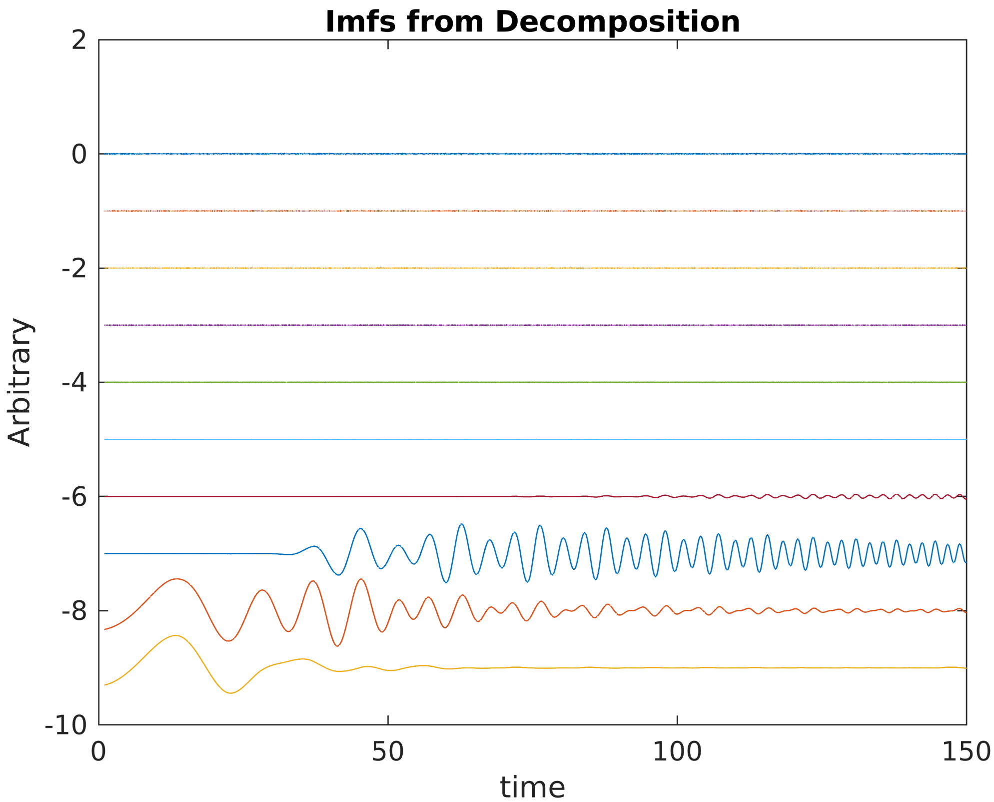

```
%plot(tspan, y_1)
%plot(tspan, y_2)
```

```
% Perform ensemble EMD
% Mitigates mode-mixing
Ne = 750;
sigma_e = 0.12; % still works with sigma = 1.5
rng(50);
[m,n] = size(y_s);

imf_eemd = eEMD(y_s, sigma_e, Ne, 10, 'pchip');
```

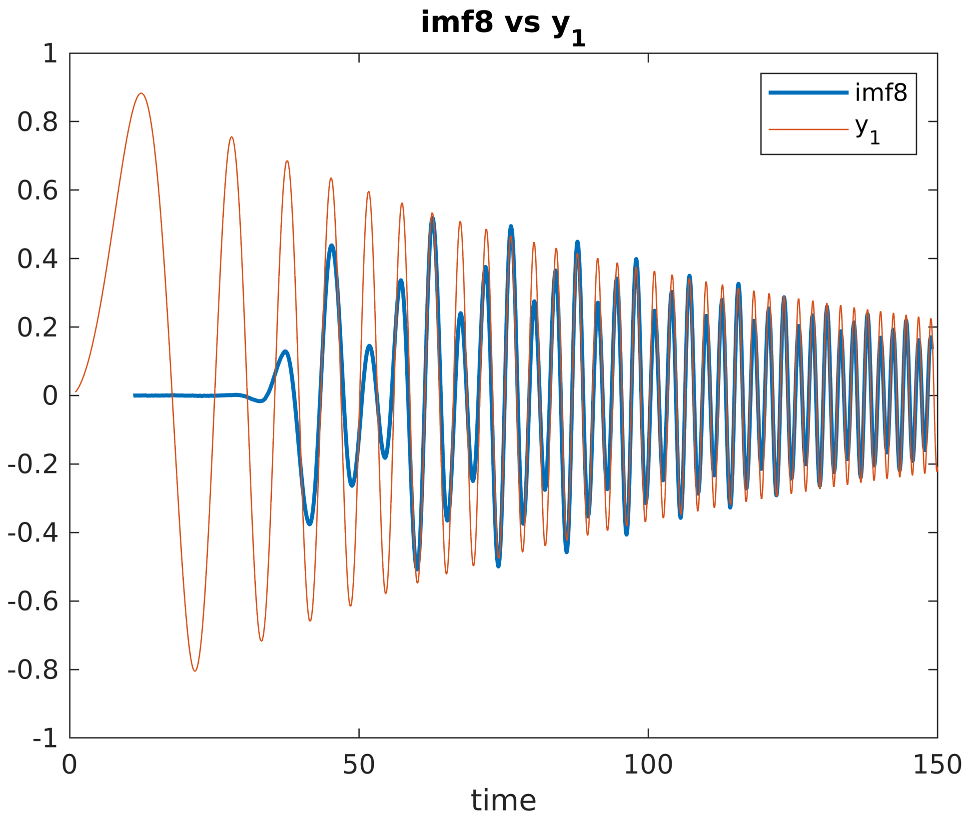
```
imf_= imf_eemd;%emd(y_s, 'Interpolation', 'spline');
[hs,f,t,imfinsf,imfinse] = hht(imf_eemd, fs/2);
```

```
clf
for i =1:1:10
    if i==1
        plot(tspan, imf_eemd(:,i))
    hold on
    end
    if i>1
        plot(tspan, imf_eemd(:,i)-1*(i-1))
```

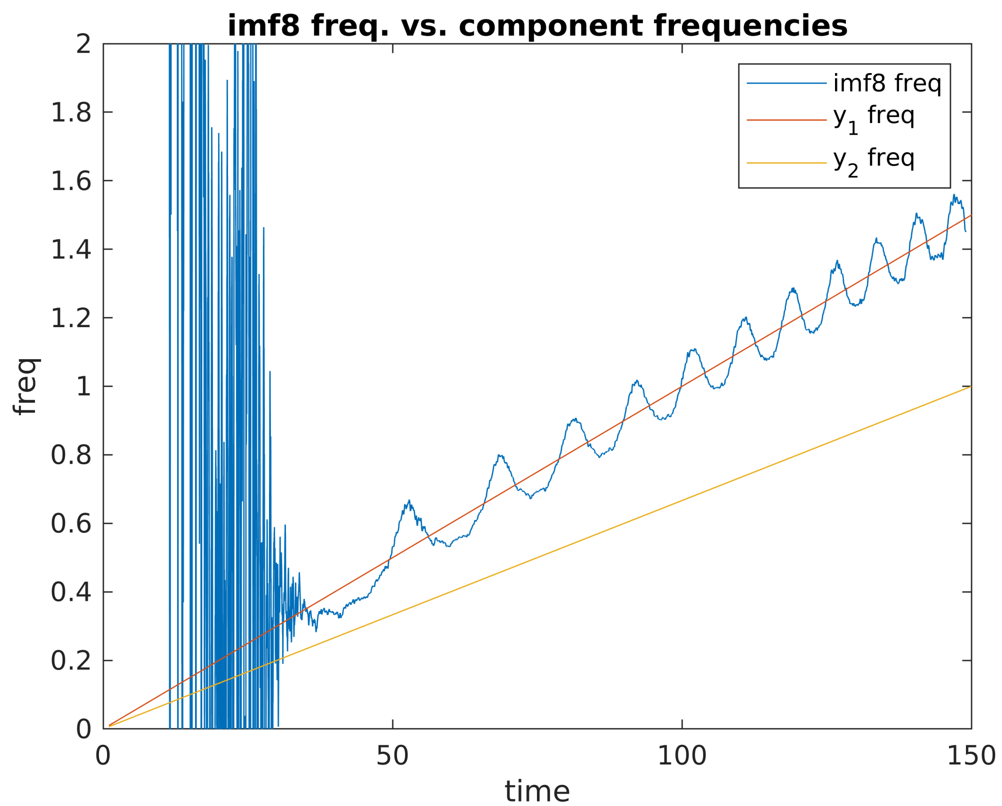
```
    end
end
title("Imfs from Decomposition")
% all of the imfs should be centred on y=0
% I shifted them to plot them all on one figure.
ylabel("Arbitrary")
xlabel("time")
```




```
clf
plot(tspan(10000:148000), imf_eemd(10000:148000,8), 'LineWidth',1.5)
hold on
plot(tspan, y_1)
title("imf8 vs y_1")
xlabel("time")
legend("imf8", "y_1")
hold off
```




```
clf
plot(tspan(10000:148000), 2.*pi.*imfinsf(10000:148000,8))
hold on
plot(tspan, (1./100).*tspan)
plot(tspan, (1./150).*tspan)
ylabel("freq")
xlabel("time")
legend("imf8 freq", "y_1 freq", "y_2 freq")
title("imf8 freq. vs. component frequencies")
ylim([0,2])
```




```
% If we fit the slope of imf8 with lin. regression:
b_imf8 = tspan\(2.*pi.*imfinsf(:,8))
```

```
b_imf8 = 0.0104
```

```
1./150
```

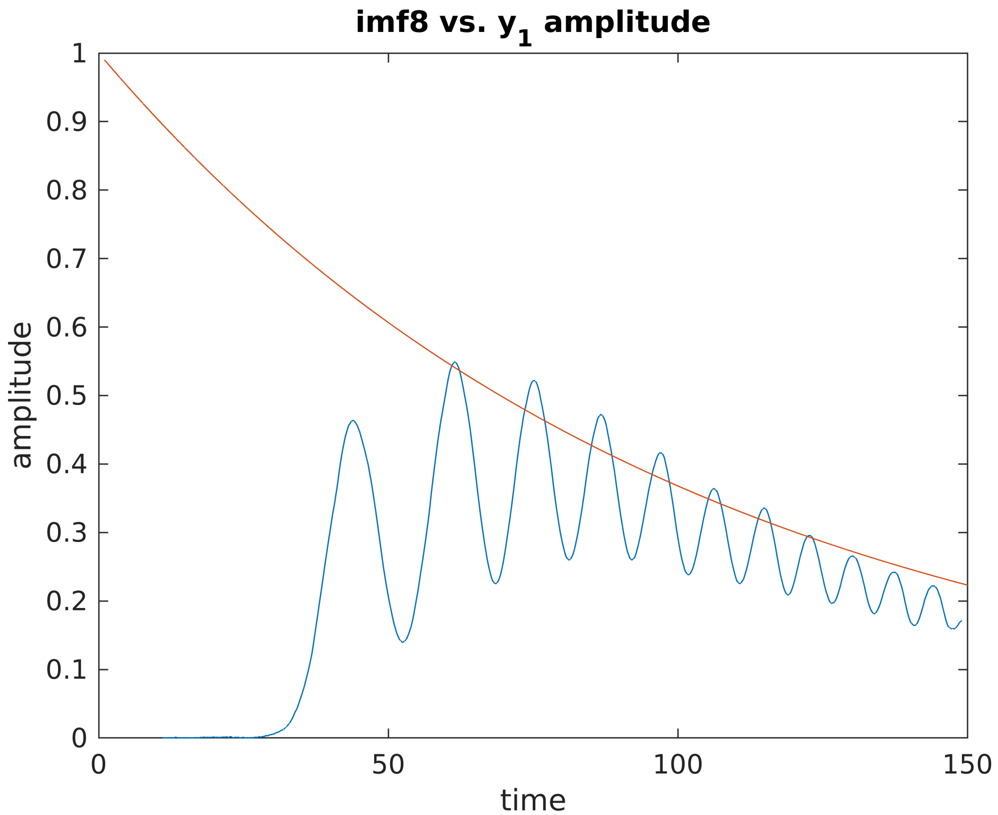
```
ans = 0.0067
```

```
1./100
```

```
ans = 0.0100
```

```
% We get very close to 1/100, the slope of freq of y_1
```

```
clf
plot(tspan(10000:148000), sqrt(imfinse(10000:148000,8)))
hold on
plot(tspan, exp(-(1./100).*tspan))
ylabel("amplitude")
xlabel("time")
title("imf8 vs. y_1 amplitude")
```




```
% Fit the amplitude assuming exponential function:
F = @(x,xdata)x(1)*exp(-x(2)*xdata);
x0 = [1, 0.1]; % educated guess
[x,resnorm,~,exitflag,output] = lsqcurvefit(F,x0,tspan(70000:148000),sqrt(imfinse(70000
```

```
Local minimum possible.

lsqcurvefit stopped because the final change in the sum of squares relative to
its initial value is less than the default value of the function tolerance.

<stopping criteria details>
x = 1×2
    0.9601    0.0110
resnorm = 215.3631
exitflag = 3
output = struct with fields:
     firstorderopt: 0.0180
        iterations: 9
         funcCount: 30
      cgiterations: 0
         algorithm: 'trust-region-reflective'
          stepsize: 8.1562e-05
           message: 'Local minimum possible.↵↵lsqcurvefit stopped because the final change in the sum of squares
```

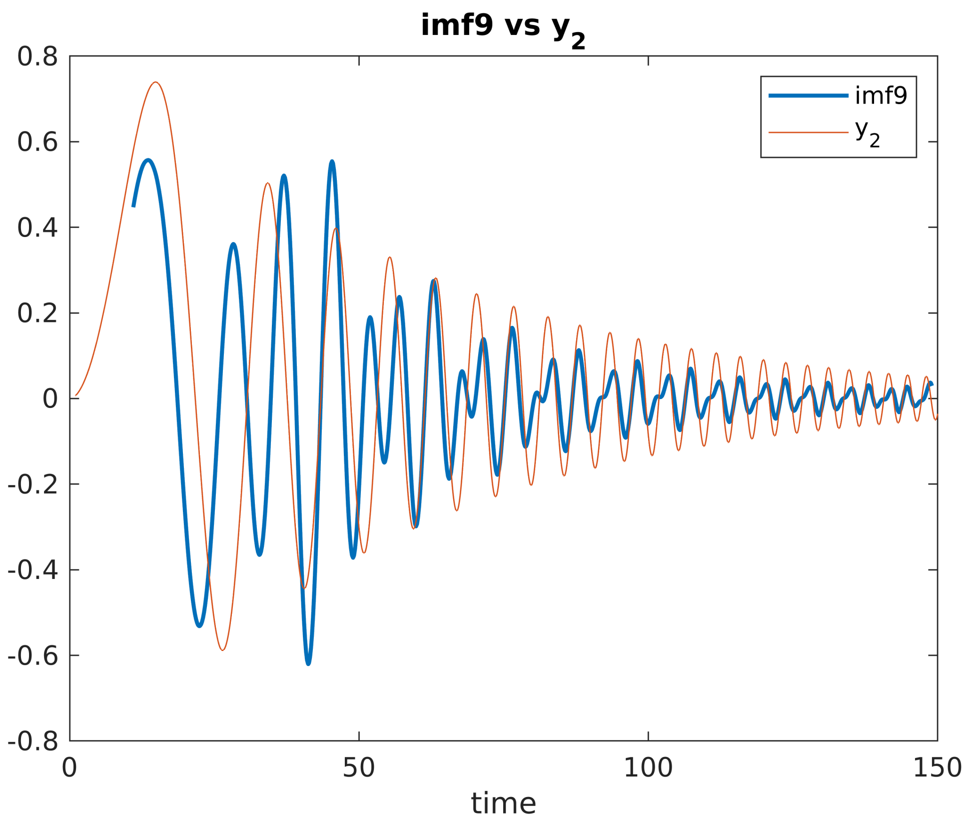
```
% Notice that I only gave data from t=70 onwards. This is kind of cheating
```

```
% We get x(1) exp(-x(2)t)
x
```

```
x = 1×2
    0.9601    0.0110
```

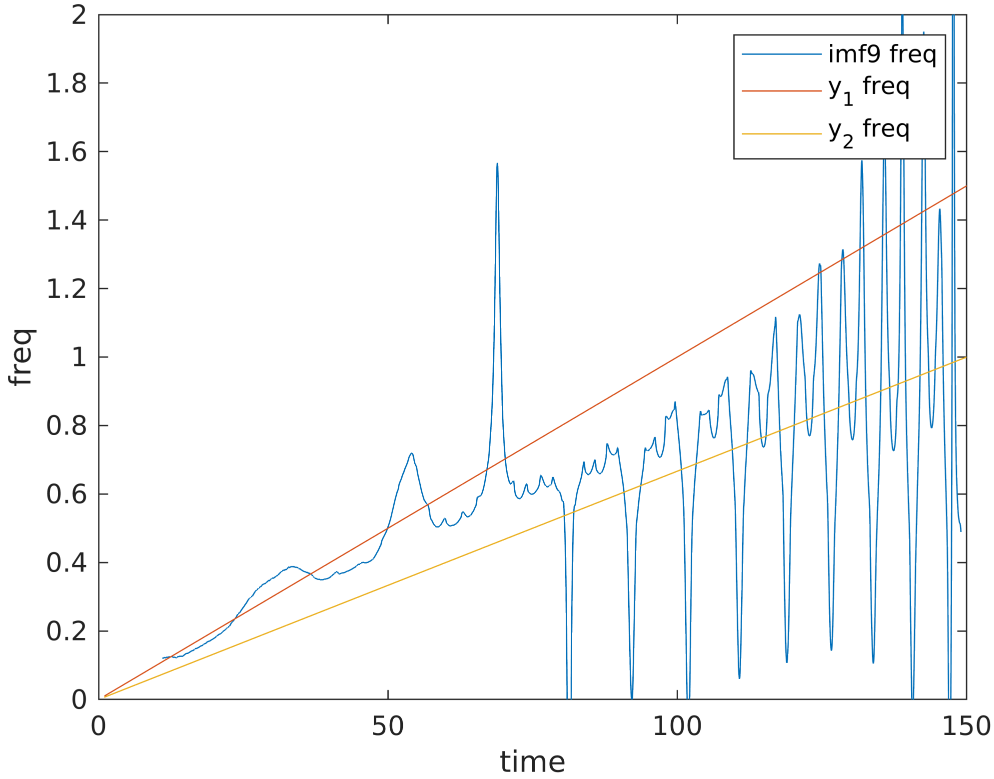
```
% x(2) matches 1/100 = 0.01 pretty well
```

```
clf
plot(tspan(10000:148000), imf_eemd(10000:148000,9), 'LineWidth',1.5)
hold on
plot(tspan, y_2)
title("imf9 vs y_2")
xlabel("time")
legend("imf9", "y_2")
hold off
```




```
clf
plot(tspan(10000:148000), 2.*pi.*imfinsf(10000:148000,9))
hold on
plot(tspan, (1./100).*tspan)
plot(tspan, (1./150).*tspan)
ylabel("freq")
xlabel("time")
```

```
legend("imf9 freq", "y_1 freq", "y_2 freq")
ylim([0,2])
```




```
% If we fit the slope of imf9 with lin. regression:
b_imf9 = tspan\(2.*pi.*imfinsf(:,9))
```

```
b_imf9 = 0.0069
```

```
1./150
```

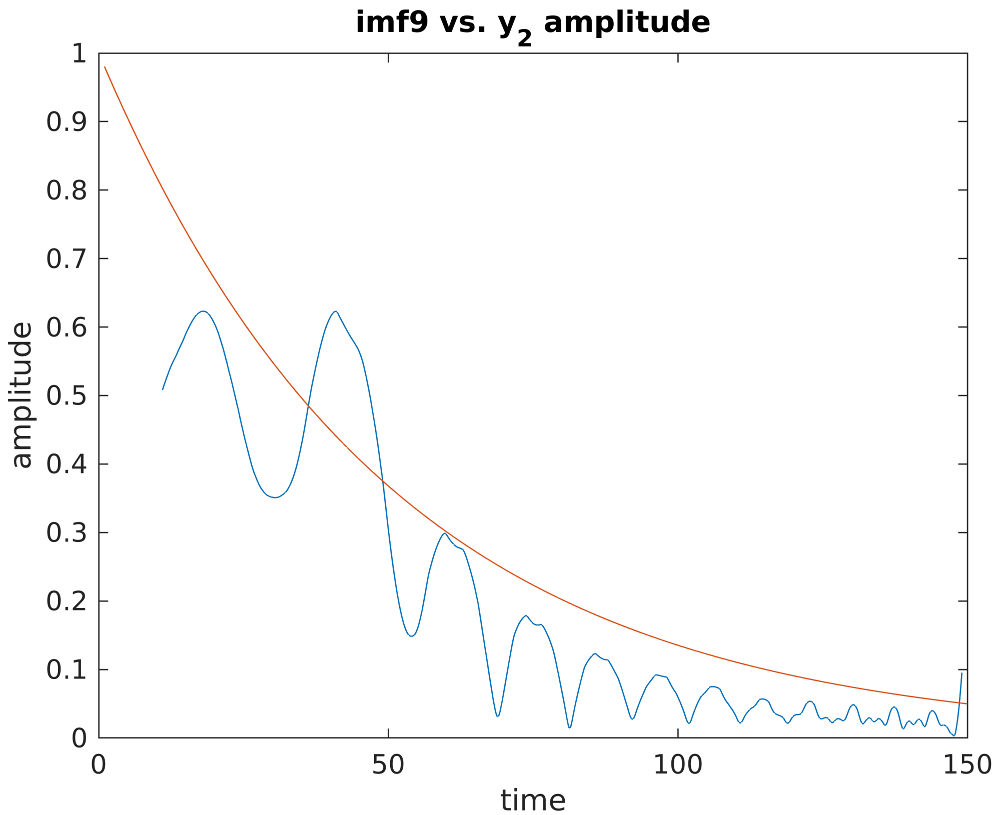
```
ans = 0.0067
```

```
1./100
```

```
ans = 0.0100
```

```
% We get very close to 1/150, the slope of freq of y_2
```

```
clf
plot(tspan(10000:148000), sqrt(imfinse(10000:148000,9)))
hold on
plot(tspan, exp(-(1./50).*tspan))
ylabel("amplitude")
xlabel("time")
title("imf9 vs. y_2 amplitude")
```




```
% Fit the amplitude assuming exponential function:
F = @(x,xdata)x(1)*exp(-x(2)*xdata);
x0 = [1, 1];
[x,resnorm,~,exitflag,output] = lsqcurvefit(F,x0,tspan(10000:148000),sqrt(imfinse(10000
```

```
Local minimum possible.

lsqcurvefit stopped because the final change in the sum of squares relative to
its initial value is less than the default value of the function tolerance.

<stopping criteria details>
x = 1×2
    0.8873    0.0225
resnorm = 970.0074
exitflag = 3
output = struct with fields:
    firstorderopt: 7.7710
       iterations: 15
        funcCount: 48
     cgiterations: 0
        algorithm: 'trust-region-reflective'
         stepsize: 2.4352e-04
          message: 'Local minimum possible.↵↵lsqcurvefit stopped because the final change in the sum of squares
```

```
% We get x(1) exp(-x(2)t)
x
```

```
x = 1×2
    0.8873    0.0225
```

```
% x(2) matches 1/50 = 0.02 pretty well
```