Differential Equations&*Mathematica*

©1999 Bill Davis and Jerry Uhl
Produced by Bruce Carpenter   Published by Math Everywhere, Inc.
www.matheverywhere.com

DE.03 Laplace Transform and Fourier Analysis Basics

B.1) The Laplace transform Y[s] of y[t] is

$$Y[s] = \int_0^{\text{infinity}} E^{-s\,t}\, y[t]\, dt.$$

The Laplace transform of y'[t] is s Y[s] – y[0].

The Laplace transform of y''[t] is s^2 Y[s] – s y[0] – y′[0]

Here's the inside scoop on the Laplace transform:
The functions legal for the Laplace transform are all functions f[t] with the property that

$E^{-s\,t}$ f[t] → 0 as t → ∞

for all large positive s.

Sin[p t], Cos[p t] , $E^{k\,t}$, Log[t]

and any quotient of polynomials are all legal.
The Laplace transform of a given function f[t] is another function, F[s], given by

$$F[s] = \int_0^{\text{infinity}} E^{-s\,t}\, f[t]\, dt.$$

(The t variable is integrated out, leaving only s).

For instance, here's the Laplace transform of
f[t] = t:

```
Clear[f, F, s, t];
f[t_] = t;
F[s_] = LaplaceTransform[f[t], t, s]
```

$\frac{1}{s^2}$

Check:

$\int_0^\infty E^{-s\,t}\, f[t]\, dt$

If$\left[\text{Re}[s] > 0, \frac{1}{s^2}, \int_0^\infty E^{-s\,t}\, t\, dt\right]$

This is Mathematica's way of telling you that if s > 0, then

$\int_0^{\text{infinity}} E^{-s\,t}\, f[t]\, dt = \frac{1}{s^2}$.

If Mathematica is given the Laplace transform of f[t], then Mathematica can often recover the formula for f[t]:

```
InverseLaplaceTransform[F[s], s, t]
t
```

Some more play:

□Experiment 1:

```
Clear[f, F, s, t];
f[t_] = Sin[t];
F[s_] = LaplaceTransform[f[t], t, s]
```

$\frac{1}{1+s^2}$

```
InverseLaplaceTransform[F[s], s, t]
Sin[t]
InverseLaplaceTransform[F[s], s, t] == f[t]
True
```

□Experiment 2:

```
Clear[f, F, s, t];
f[t_] = E^-t;
F[s_] = LaplaceTransform[f[t], t, s]
```

$\frac{1}{1+s}$

```
InverseLaplaceTransform[F[s], s, t]
```

E^{-t}

```
InverseLaplaceTransform[F[s], s, t] == f[t]
True
```

□Experiment 3:

```
Clear[f, F, s, t];
F[s_] = 2/(s^2 + 4)
```

$\frac{2}{4+s^2}$

```
f[t_] = InverseLaplaceTransform[F[s], s, t]
Sin[2 t]
LaplaceTransform[f[t], t, s]
```

$\frac{2}{4+s^2}$

□Experiment 4:

```
Clear[f, Lf, s, x];
F[s_] = 5.2/(s + 2)
```

$\frac{5.2}{2+s}$

```
f[t_] = InverseLaplaceTransform[F[s], s, t]
```

$5.2\,E^{-2\,t}$

```
LaplaceTransform[f[t], t, s]
```

$\frac{5.2}{2+s}$

Before computers, folks did these by looking up functions and their inverse Laplace transforms in printed tables.

Different functions have different Laplace transforms.

□B.1.a.i) The Laplace transform of the first derivative

Look at this:

```
Clear[y, t, s];
LaplaceTransform[y′[t], t, s]
s LaplaceTransform[y[t], t, s] - y[0]
```

One reason folks fall for the charms of the Laplace transform is that if
g[t] = y'[t],
then you are guaranteed that the Laplace transforms line up like this:
G[s] = s Y[s] – y[0]
where
G[s] is the Laplace transform of g[t] = y′[t]
and

Y[s] is the Laplace transform of y[t].

This is a good fringe benefit because it tells you how to write down the Laplace transform of g[t] = y′[t] once you've got your hands on y[t] and its Laplace transform.
Play with it:

□Experiment 1:

```
Clear[y, Y, s, t];
y[t_] = E^(-0.4 t);
LaplaceTransform[y′[t], t, s]
```

$-\frac{0.4}{0.4+s}$

```
Together[s LaplaceTransform[y[t], t, s] - y[0]]
```

$-\frac{0.4}{0.4+s}$

□Experiment 2:

```
Clear[y, s, t];
y[t_] = E^-t Cos[π t];
Together[LaplaceTransform[y′[t], t, s]]
```

$\frac{-1-\pi^2-s}{1+\pi^2+2\,s+s^2}$

```
Together[s LaplaceTransform[y[t], t, s] - y[0]]
```

$\frac{-1-\pi^2-s}{1+\pi^2+2\,s+s^2}$

Even when Mathematica can't handle the calculations, you can be certain that if
g[t] = y′[t],
then you are guaranteed that
G[s] = s Y[s] – y[0]
because you can use integration by parts to explain this formula. Do it.

□Answer:

Remember,

$$Y[s] = \int_0^\infty E^{-s\,t}\, y[t]\, dt,$$

and because g[t] = y′[t]

$$G[s] = \int_0^\infty E^{-s\,t}\, y'[t]\, dt, .$$

The upshot:

Explaining why

$$G[s] = s\,Y[s] - y[0]$$

is the same as explaining why

$$\int_0^\infty E^{-s\,t}\,y'[t]\,dt = s\int_0^\infty E^{-s\,t}\,y[t]\,dt - y[0].$$

To get a grasp on the left side, remember

$$\int_0^\infty E^{-s\,t}\,y'[t]\,dt = \text{limit of } \int_0^n E^{-s\,t}\,y'[t]\,dt \text{ as } n \to \infty$$

Now work on

$$\int_0^n E^{-s\,t}\,y'[t]\,dt$$

using integration by parts:

The integration by parts formula is

$$\int_0^n u[t]\,v'[t]\,dt = u[t]\,v[t]\,\Big|_0^n - \int_0^n v[t]\,u'[t]\,dt.$$

Make the assignments:

$$u[t] = E^{-s\,t} \text{ and } v'[t] = y'[t].$$

This gives

$$u'[t] = -s\,E^{-s\,t} \text{ and } v[t] = y[t]$$

Plug into the integration by parts formula to get

$$\int_0^n E^{-s\,t}\,y'[t]\,dt$$

$$= E^{-s\,t}\,y[t]\Big|_0^n - \int_0^n -s\,E^{-s\,t}\,y[t]\,dt$$

$$= E^{-s\,t}\,y[t]\Big|_0^n + s\int_0^n E^{-s\,t}\,y[t]\,dt$$

$$= E^{-s\,n}\,y[n] - y[0] + s\int_0^n E^{-s\,t}\,y[t]\,dt.$$

As $n \to \infty$,

$$\int_0^n E^{-s\,t}\,y'[t]\,dt \to G[s],$$

$$E^{-s\,n}\,y[n] \to 0,$$

and

$$s\int_0^n E^{-s\,t}\,y[t]\,dt \to s\,Y[s].$$

So

$$G[s] = -y[0] + s\,Y[s] = s\,Y[s] - y[0]$$

as advertised.

If someone ever asks you whether you have seen the Laplace transform, now you can say that you've seen a little bit about it. And you can tell them that you saw it first in DiffEq&*Mathematica*.

□B.1.a.ii)

Look at this:

```
Clear[y, t, s];
LaplaceTransform[y''[t], t, s]
```

$$s^2\,\text{LaplaceTransform}[y[t], t, s] - s\,y[0] - y'[0]$$

You already know the Laplace transform of y′[t] is

$$s\,Y[s] - y[0].$$

How does this fact guarantee that the Laplace transform of y″[t] is

$$s^2\,Y[s] - s\,y[0] - y'[0]?$$

□Answer:

Put

$$h[t] = y''[t] \text{ and } g[t] = y'[t]$$

so that

$$h[t] = g'[t].$$

According to part a.i) above, you are guaranteed that

$$H[s] = s\,G[s] - g[0]:$$

```
Clear[G, H, g, y, Derivative, s];
H[s_] = s G[s] - y'[0]
```

```
s G[s] - y'[0]
```

The formula in part a.i) also says that

$$G[s] = s\,Y[s] - y[0]:$$

```
Expand[H[s] /. G[s] → s Y[s] - y[0]]
```

$$-s\,y[0] + s^2\,Y[s] - y'[0]$$

So

$$H[s] = s^2\,Y[s] - s\,y[0] - y'[0],$$

as advertised.

Try this formula out:

□Experiment 1:

```
Clear[y, s, x, t];
y[t_] = Sin[5 t];
LaplaceTransform[y''[t], t, s]
```

$$-\frac{125}{25 + s^2}$$

$$\text{Together}[s^2\,\text{LaplaceTransform}[y[t], t, s] - s\,y[0] - y'[0]]$$

$$-\frac{125}{25 + s^2}$$

□Experiment 2:

$$\begin{aligned}&\text{Clear}[y, s, t];\\ &y[t\_] = \text{Sin}[t] + \frac{1}{2}\,\text{Sin}[2\,t];\\ &\text{Together}[\text{LaplaceTransform}[y''[t], t, s]]\end{aligned}$$

$$\frac{-8 - 5\,s^2}{(1 + s^2)\,(4 + s^2)}$$

$$\text{Together}[s^2\,\text{LaplaceTransform}[y[t], t, s] - s\,y[0] - y'[0]]$$

$$\frac{-8 - 5\,s^2}{(1 + s^2)\,(4 + s^2)}$$

It works in theory even in cases *Mathematica* can't handle.

B.2) Using the magic of the Laplace Transform to come up with formulas for forced exponential diffeqs and forced oscillator diffeqs

□B.2.a) Using the Laplace transform to solve a forced exponential diffeq

Showcase the Laplace transform by using it to come up with the exact formula for the solution of

$$y'[t] + 1.2\,y[t] = 4.75\,E^{-0.09\,t}\,\text{Cos}[3.5\,t]$$

with $y[0] = 5.2$.

□Answer:

With the Laplace transform, this is duck soup. You look at the diffeq:

$$y'[t] + 1.2\,y[t] = 4.75\,E^{-0.09\,t}\,\text{Cos}[3.5\,t]$$

with $y[0] = 5.2$.

and then you enter

$$\begin{aligned}&\text{Clear}[f, t, Y, y, s];\\ &r = 1.2;\\ &f[t\_] = 4.75\,E^{-0.09\,t}\,\text{Cos}[3.5\,t];\\ &\text{starter} = 5.2;\ Y[s\_] = \frac{\text{LaplaceTransform}[f[t], t, s] + \text{starter}}{r + s}\end{aligned}$$

$$\frac{5.2 + \frac{4.75\,(0.09+s)}{12.25+(0.09+s)^2}}{1.2 + s}$$

This is the Laplace transform of the solution.

To get the formula for the solution y[t] you calculate the inverse Laplace transform of Y[s].

The formula for the solution y[t] is:

```
y[t_] = Chop[Expand[InverseLaplaceTransform[Y[s], s, t]]]
```

$$4.80893\,E^{-1.2\,t} + 0.391074\,E^{-0.09\,t}\,\text{Cos}[3.5\,t] + 1.23312\,E^{-0.09\,t}\,\text{Sin}[3.5\,t]$$

That's all there is to it.

Check whether this y[t] solves $y'[t] + 1.2\,y[t] = 4.75\,E^{-0.09\,t}\,Cos[3.5\,t]$:

```
Chop[Expand[y′[t] + 1.2 y[t]]]
4.75 E^(-0.09 t) Cos[3.5 t]
```

Yepper.

Check whether this y[0] = 5.2 for this y[t]:

```
y[0]
5.2
```

Everything checks.

□B.2.b) Using the Laplace transform to solve a forced oscillator diffeq

Showcase the Laplace transform by using it to come up with the exact formula for the forced, damped oscillator coming from

$$y''[t] + 1.3\,y'[t] + 6.5\,y[t] = 9.1\,E^{-0.8\,t}$$

with y[0] = 1.6 and y′[0] = 3.7.

□ **Answer:**

With the Laplace transform, this is also duck soup.

You look at the diffeq:

$$y''[t] + 1.3\,y'[t] + 6.5\,y[t] = 9.1\,E^{-0.8\,t}$$

with y[0] = 1.6 and y′[0] = 3.7.

and then you enter

```
Clear[f, t, Y, y, s]
b = 1.3; c = 6.5;
f[t_] = 9.1 E^(-0.8 t);
ystarter = 5.2; yprimestarter = 3.7;
Y[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] + b ystarter +
    s ystarter + yprimestarter)
(10.46 + 5.2 s + 9.1/(0.8+s))/(6.5 + 1.3 s + s^2)
```

This is the Laplace transform of the solution.

To get the formula for the solution y[t] you calculate the inverse Laplace transform of Y[s].

The formula for the solution y[t] is:

```
y[t_] = Chop[Expand[InverseLaplaceTransform[Y[s], s, t]]]
1.4918 E^(-0.8 t) + 3.7082 E^(-0.65 t) Cos[2.46526 t] + 2.96268 E^(-0.65 t) Sin[2.46526 t]
```

That's all there is to it.

Check whether this y[t] solves $y''[t] + 1.3\,y'[t] + 6.5\,y[t] = 9.1\,E^{-0.8\,t}$:

```
Chop[Expand[(y′)′[t] + 1.3 y′[t] + 6.5 y[t]]]
9.1 E^(-0.8 t)
```

Yepper.

Check whether y[0] = 1.6 and y′[0] = 3.7.

```
y[0]
5.2
y'[0]
3.7
```

Everything checks.

Amazing thing this Laplace transform.

□B.2.c) Using the Laplace transform to solve a forced oscillator diffeq

You are using Mathematica to handle the Laplace transform.
How was it done in your grandparents' diffeq class?

□ **Answer:**

In your grandparents' time, they would look at a diffeq such as

$$y''[t] + 2.0\,y'[t] + 6.5\,y[t] = 9.1\,E^{-1.9\,t}$$

with y[0] = 1.5 and y′[0] = −4.7.

and write down

$$Y[s] = \frac{\text{LaplaceTransform}[f[t],t,s] + b\ \text{ystarter} + s\ \text{ystarter} + \text{yprimestarter}}{s^2 + b\,s + c}$$

with b = 2.0, c = 6.5, $f[t] = 9.1\,E^{-1.9\,t}$, ystarter = 1.5 and yprimestarter = −4.7.

Then your grandparents would look in a printed table to find which function y[t] has Y[s] for its Laplace transform.

This is similar to what you are doing, but you're using the computer to find which function y[t] has Y[s] for its Laplace transform.

□B.2.d)

Part of the job of mathematics is to explain why things are guaranteed to work out.
Explain why the methods used in parts a) and b) above worked.

□ **Answer:**

□Explanation for the forced exponential diffeq

y'[t] + r y[t] = f[t]:

It's just a little algebra.

Enter the differential equation with everything cleared:

```
Clear[y, t, f, Y, r, s];
diffeq = y'[t] + r y[t] == f[t]
r y[t] + y′[t] == f[t]
```

Replace all the functions involved by their Laplace transforms:

```
laplaced = diffeq /.
    {y[t] -> LaplaceTransform[y[t], t, s],
     y'[t] -> LaplaceTransform[y'[t], t, s],
     f[t] -> LaplaceTransform[f[t], t, s]}
r LaplaceTransform[y[t], t, s] + s LaplaceTransform[y[t], t, s] - y[0] ==
 LaplaceTransform[f[t], t, s]
```

Solve this for the Laplace Transform of y[t]:

```
sol = Simplify[Solve[laplaced, LaplaceTransform[y[t], t, s]]]
{{LaplaceTransform[y[t], t, s] → (LaplaceTransform[f[t], t, s] + y[0])/(r + s)}}
```

There is is.

This tells you that the Laplace transform Y[s] of the solution y[t] is

$$Y[s] = \frac{\text{LaplaceTransform}[f[t],t,s] + y[0]}{r + s}$$

which is exactly the formula used in part a).

□Explanation for the forced oscillator diffeq

y''[t] + b y'[t] + c y[t] = f[t]:

Again, it's just a little algebra.

Enter the differential equation with everything cleared:

```
Clear[y, t, f, b, c, s, Y];
diffeq = y''[t] + b y'[t] + c y[t] == f[t]
c y[t] + b y′[t] + y″[t] == f[t]
```

Replace all the functions involved by their Laplace transforms:

```
laplaced = diffeq /.
    {y[t] -> LaplaceTransform[y[t], t, s],
     y'[t] -> LaplaceTransform[y'[t], t, s],
     y''[t] -> LaplaceTransform[y''[t], t, s],
     f[t] -> LaplaceTransform[f[t], t, s]}
c LaplaceTransform[y[t], t, s] + s^2 LaplaceTransform[y[t], t, s] +
  b (s LaplaceTransform[y[t], t, s] - y[0]) - s y[0] - y′[0] ==
 LaplaceTransform[f[t], t, s]
```

Solve this for the Laplace Transform of y[t]:

```
sol = Solve[laplaced, LaplaceTransform[y[t], t, s]]
{{LaplaceTransform[y[t], t, s] →
   -(-LaplaceTransform[f[t], t, s] - b y[0] - s y[0] - y′[0])/(c + b s + s^2)}}
```

There is is.

This tells you that the Laplace transform Y[s] of the solution y[t] is

$$Y[s] = \frac{\text{LaplaceTransform}[f[t],t,s] + b\,y[0] + s\,y[0] + y'[0]}{c + b\,s + s^2}$$

which is exactly the formula used in part b).

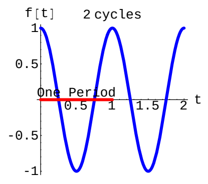
B.3) Fourier fit of periodic functions

□B.3.a)

Look at these:

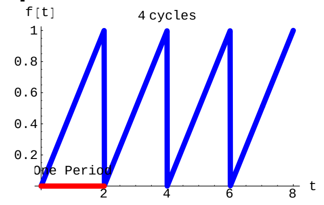
□Function 1:

```
Clear[f, t];
f[t_] = Cos[2 π t];
L = 1;
cycles = 2;
Plot[f[t], {t, 0, cycles L}, PlotStyle → {{Thickness[0.02], Blue}},
 AxesLabel → {"t", "f[t]"}, PlotLabel → "cycles" cycles,
 Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
   {Text["One Period", {L/2, 0.1}]}}];
```



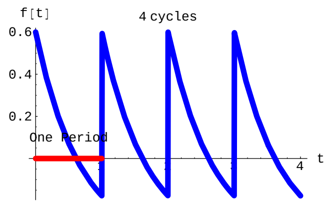
□Function 2:

```
Clear[f, t];
f[t_] = t/2 - Floor[t/2];
L = 2;
cycles = 4;

Plot[f[t], {t, 0, cycles L},
 PlotStyle → {{Thickness[0.02], Blue}}, AxesLabel → {"t", "f[t]"},
 PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
 Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
   {Text["One Period", {L/2, 0.1}]}}];
```



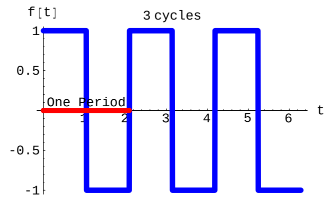
□Function 3:

```
Clear[f, t];
f[t_] = -0.4 + E^(-1.5 (t-Floor[t]));
L = 1;
cycles = 4;

Plot[f[t], {t, 0, cycles L},
 PlotStyle → {{Thickness[0.02], Blue}}, AxesLabel → {"t", "f[t]"},
 PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
 Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
   {Text["One Period", {L/2, 0.1}]}}];
```



□Function 4:

```
Clear[f, t];
f[t_] = Sign[Sin[3 t]];
L = 2π/3;
cycles = 3;

Plot[f[t], {t, 0, cycles L},
 PlotStyle → {{Thickness[0.02], Blue}}, AxesLabel → {"t", "f[t]"},
 AspectRatio → 1/GoldenRatio, PlotLabel → "cycles" cycles,
 Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
   {Text["One Period", {L/2, 0.1}]}}];
```



Folks call all of these functions periodic.
How do you recognize periodic functions?

□Answer:

Most folks, like you, just eyeball a plot looking for cycles.

Once you spot a cycle, then you can say that its length L represents one period of the function. This gives you

$$f[t + L] = f[t] \text{ for all } t's.$$

This is the definition that some folks prefer when they talk about periodic functions.

□B.3.b.i) Fast Fourier fit of periodic functions

At the December, 1807 meeting of the French Academy of Sciences, mathematician and engineer Joseph Fourier announced that you can approximate an arbitrary function by sums of sine and cosine waves. Those in attendance thought he was slightly off his rocker, but eventually they were all forced to change their minds because time has confirmed that Fourier's idea was one of the great scientific break-throughs of all time. In this problem, you get the opportunity to get your hands on Fourier's idea in action.
Go for it.

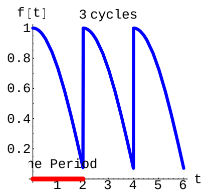
Activate these custom *Mathematica* instructions made for your enjoyment and economic use.

They come from the home office to you.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
  jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];


Fourierfitters[L_, n_, t_] := Table[E^(2πIkt/L),
   {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
 Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]

FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

To get an idea of what these new instructions can do for you, look at this plot of a function f[t] whose period is L = 2, plotted for three cycles:

```
Clear[f, t];
f[t_] = Cos[1.5 (t/2 - Floor[t/2])];
L = 2;
cycles = 3;
fplot =
 Plot[f[t], {t, 0, cycles L}, PlotStyle → {{Thickness[0.02], Blue}},
  AxesLabel → {"t", "f[t]"}, PlotLabel → "cycles" cycles,
  Epilog → {{Red, Thickness[0.03], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}},
  AspectRatio → 1];
```



Now stay with the same function f[t]:

```
f[t]
Cos[1.5 (t/2 - Floor[t/2])]
```

Stay with L = 2 and go with n = 4 and look at:

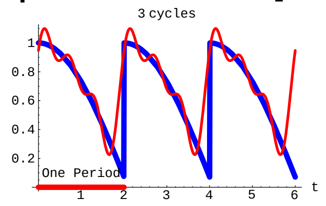
```
n = 4;
FastFourierfit[f, L, n, t]
0.721126 + (0.0158769 + 0.149139 I) E^(-I π t) + (0.0158769 - 0.149139 I) E^(I π t) +
 (0.0462506 + 0.059115 I) E^(-2 I π t) + (0.0462506 - 0.059115 I) E^(2 I π t) +
 (0.0512009 + 0.0243066 I) E^(-3 I π t) + (0.0512009 - 0.0243066 I) E^(3 I π t)
```

That ran really fast.
Now look at this plot of f[t] and FastFourierfit[f, L, n, t] with
n = 4 and L = 2:

```
Clear[realfit];
realfit[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
0.721126 + 0.0317538 Cos[π t] + 0.0925012 Cos[2 π t] + 0.102402 Cos[3 π t] +
 0.298279 Sin[π t] + 0.11823 Sin[2 π t] + 0.0486133 Sin[3 π t]
```

See how realfit[t] plots out:

```
fitplot = Plot[{f[t], realfit[t]}, {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  AxesLabel → {"t", ""},
  PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```

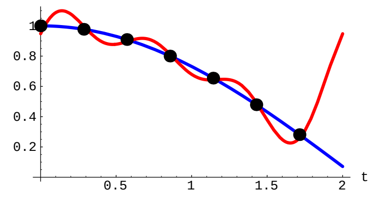


What's happening here?

□ **Answer:**

Look at this embellished plot of the first cycle still going with L = 2 and n = 4:

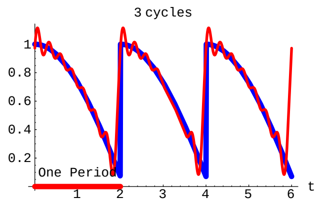
```
first = Plot[{f[t], realfit[t]}, {t, 0, L},
  PlotStyle → {{Thickness[0.01], Blue}, {Thickness[0.01], Red}},
  AxesLabel → {"t", ""}, DisplayFunction → Identity];
fdata = Table[N[{t, f[t]}], {t, 0, L - L/(2 n - 1), L/(2 n - 1)}];
fdataplot = ListPlot[fdata,
  PlotStyle → PointSize[0.04], DisplayFunction → Identity];
Show[first, fdataplot, DisplayFunction → $DisplayFunction];
```



Evidently, Fourierfit[f, n, t] picks 2 n – 1 equally spaced data points off the plot of f[t] between t = 0 and t = L and tries to fit these points with a combination of complex exponentials.

When you kick n up, you get a really good fit:

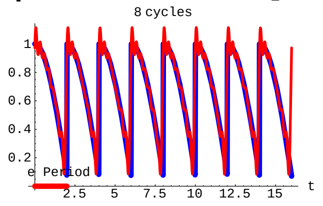
```
n = 8;
FastFourierfit[f, L, n, t]
0.693549 - (0.0116423 - 0.154929 I) E^(-I π t) - (0.0116423 + 0.154929 I) E^(I π t) +
 (0.0189249 + 0.0711746 I) E^(-2 I π t) + (0.0189249 - 0.0711746 I) E^(2 I π t) +
 (0.0242786 + 0.043772 I) E^(-3 I π t) + (0.0242786 - 0.043772 I) E^(3 I π t) +
 (0.0261085 + 0.0291675 I) E^(-4 I π t) + (0.0261085 - 0.0291675 I) E^(4 I π t) +
 (0.0269223 + 0.0194654 I) E^(-5 I π t) + (0.0269223 - 0.0194654 I) E^(5 I π t) +
 (0.0273257 + 0.0120596 I) E^(-6 I π t) + (0.0273257 - 0.0120596 I) E^(6 I π t) +
 (0.0275192 + 0.00578952 I) E^(-7 I π t) + (0.0275192 - 0.00578952 I) E^(7 I π t)
Clear[realfit];
realfit[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
0.693549 - 0.0232846 Cos[π t] + 0.0378498 Cos[2 π t] +
 0.0485571 Cos[3 π t] + 0.052217 Cos[4 π t] + 0.0538446 Cos[5 π t] +
 0.0546514 Cos[6 π t] + 0.0550385 Cos[7 π t] + 0.309858 Sin[π t] +
 0.142349 Sin[2 π t] + 0.087544 Sin[3 π t] + 0.0583351 Sin[4 π t] +
 0.0389307 Sin[5 π t] + 0.0241191 Sin[6 π t] + 0.011579 Sin[7 π t]
cycles = 3;
fitplot = Plot[{f[t], realfit[t]}, {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  AxesLabel → {"t", ""},
  PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



Hot plot.

And you get an extra benefit:
When you put the period L of f[t] into the FastFourierFit[f, L, t, n] instruction, then the periods of the complex exponentials used by the fit match the period of the forcing function f[t], and you continue to get that good fit all the way:

```
cycles = 8;
fitplot = Plot[{f[t], realfit[t]}, {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  AxesLabel → {"t", ""},
  PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



You've got to agree that Fourier knew what he was talking about.

□B.3.b.ii)

You could use the Mathematica Fit instruction to get the same effect as the FastFourierfit instruction.
Why would anyone want this new FastFourierfit instruction?

□**Answer:**

When you try to fit using many data points, the `Fit` instruction can bog down badly.

On the other hand, the `FastFourierfit` instruction really smokes.

Try it out on another function whose period is 2π.

```
Clear[f, n, t]
f[t_] = E^Sin[t];
FastFourierfit[f, 2 π, 4, t]
```

```
1.26607 + 0.565161 I E^(-I t) - 0.565161 I E^(I t) - 0.13577 E^(-2 I t) - 0.13577 E^(2 I t) -
 0.0224399 I E^(-3 I t) + 0.0224399 I E^(3 I t)
```

```
FastFourierfit[f, 2 π, 8, t]
```

```
1.26607 + 0.565159 I E^(-I t) - 0.565159 I E^(I t) - 0.135748 E^(-2 I t) - 0.135748 E^(2 I t) -
 0.0221684 I E^(-3 I t) + 0.0221684 I E^(3 I t) + 0.00273712 E^(-4 I t) + 0.00273712 E^(4 I t) +
 0.000271463 I E^(-5 I t) - 0.000271463 I E^(5 I t) - 0.0000224889 E^(-6 I t) -
 0.0000224889 E^(6 I t) - 1.60474 × 10^-6 I E^(-7 I t) + 1.60474 × 10^-6 I E^(7 I t)
```

```
FastFourierfit[f, 2 π, 12, t]
```

```
1.26607 + 0.565159 I E^(-I t) - 0.565159 I E^(I t) - 0.135748 E^(-2 I t) - 0.135748 E^(2 I t) -
 0.0221684 I E^(-3 I t) + 0.0221684 I E^(3 I t) + 0.00273712 E^(-4 I t) + 0.00273712 E^(4 I t) +
 0.000271463 I E^(-5 I t) - 0.000271463 I E^(5 I t) - 0.0000224887 E^(-6 I t) -
 0.0000224887 E^(6 I t) - 1.59922 × 10^-6 I E^(-7 I t) + 1.59922 × 10^-6 I E^(7 I t) +
 9.96062 × 10^-8 E^(-8 I t) + 9.96062 × 10^-8 E^(8 I t) + 5.51839 × 10^-9 I E^(-9 I t) -
 5.51839 × 10^-9 I E^(9 I t) - 2.75296 × 10^-10 E^(-10 I t) - 2.75296 × 10^-10 E^(10 I t)
```

Almost instant responses.

The demands of modern science made it essential to be able to do Fourier fits as fast as possible. In fact, fast Fourier fits continue to play the starring role in most everything electronic including hot fields such as signal analysis, data compression and high definition television.

The ideas behind the workings of a fast Fourier fit instruction have attracted the attention of some of the best mathematical scientists of all time. None other than Gauss came up with a fast Fourier fit algorithm in 1866. The modern version of the workings behind the fast Fourier fit sprang out of the 1960's with the work by John Tukey and his cohorts at IBM and Princeton.

In their book, "Numerical Methods and Software" (Prentice-Hall, 1989),
David Kahner, Cleve Moler and Stephen Nash say
"Many knowledgeable people feel that [Tukey's fast Fourier fit] is
the single most important contribution to computing since
the advent of the stored programming concept."

Those who want to study Tukey's idea should look at this book.

Mathematica's workings behind the `FastFourierfit` instruction depend heavily on Tukey's idea.

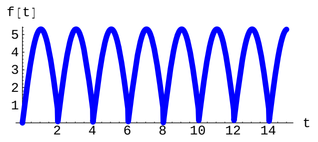
B.4) Combining Fourier fit and the Laplace transform to come up with good approximate formulas for periodically forced oscillators

"A cumbersome [exact] formula that gives a solution in explicit form is
frequently less useful than a simple approximate formula."
--------DiffEq master V. I. Arnold

□B.4.a)

You are given this periodic forcing function f[t]:

```
endtime = 15.0;
Clear[f, t]
f[t_] = 5.3 Sin[3 (t/2 - Floor[t/2])];
fplot = Plot[f[t], {t, 0, endtime},
  PlotStyle → {{Thickness[0.02], Blue}}, AxesLabel → {"t", "f[t]"}];
```



And you are asked to come up with a formula for the solution of

$$y''[t] + 1.8\,y'[t] + 8.3\,y[t] = f[t]$$

with $y[0] = 4.7$ and $y'[0] = -0.9$.

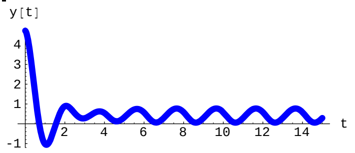
First you enter everything:

```
b = 1.8;
c = 8.3;
startery = 4.7;
starteryprime = -0.9;
Clear[t, y];

oscdiffeq = {y''[t] + b y'[t] + c y[t] == f[t],
  y[0] == startery, y'[0] == starteryprime};
ColumnForm[oscdiffeq]
```

```
8.3 y[t] + 1.8 y'[t] + y''[t] == 5.3 Sin[3 (t/2 - Floor[t/2])]
y[0] == 4.7
y'[0] == -0.9
```

Then you see how the solution plots out:

```
ndsol = NDSolve[oscdiffeq, y[t], {t, 0, endtime}, MaxSteps → 2000];
Clear[ysol];

ysol[t_] = y[t] /. ndsol[[1]]; ndsplot =
 Plot[ysol[t], {t, 0, endtime}, PlotStyle → {{Thickness[0.02], Blue}},
  PlotRange → All, AxesLabel → {"t", "y[t]"}];
```



About what you expected.
Now the job is to come up with a formula for this solution.
You try the convolution integral method and Mathematica bogs down.
The integrals are just too complicated.

Then you think of the LapalceTransform and experiment with the Laplace transform of f[t]:

```
LaplaceTransform[f[t], t, s]
```

```
5.3 LaplaceTransform[Sin[3 (t/2 - Floor[t/2])], t, s]
```

Mathematica again balks.
The integrals involved are just to complicated.
What do you do?

□**Answer:**

Instead of solving

$$y''[t] + 1.8\,y'[t] + 8.3\,y[t] = f[t]$$

with $y[0] = 4.7$ and $y'[0] = -0.9$,

you use Fourier fit to come up with a good sine and cosine wave approximation, approxf[t], of f[t] and then you use the Laplace Transform to solve the nearby oscillator diffeq

$$y''[t] + 1.8\,y'[t] + 8.3\,y[t] = \text{approxf}[t]$$

with $y[0] = 4.7$ and $y'[0] = -0.9$.

Here you go:

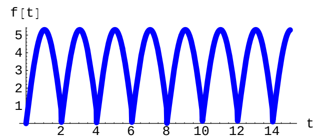
Activate this code:

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];


Fourierfitters[L_, n_, t_] := Table[E^((2 π I k t)/L),
     {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
 Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]
```

```
FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

Look at

```
Show[fplot];
```

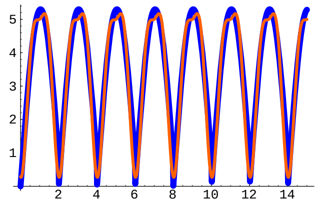


The period of this function is L = 2.

Get a good Fourier fit of f[t]:

```
L = 2;
n = 3;
Clear[approxf];
approxf[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
fitplot = Plot[{f[t], approxf[t]}, {t, 0, endtime}, PlotStyle ->
    {{Thickness[0.02], Blue}, {Thickness[0.01], CadmiumOrange}},
   AspectRatio -> 1/GoldenRatio];
```

$$3.37978 - 2.35661\,\mathrm{Cos}[\pi t] - 0.736414\,\mathrm{Cos}[2\pi t] - 0.285912\,\mathrm{Sin}[\pi t] - 0.0783657\,\mathrm{Sin}[2\pi t]$$



Nice fit:

Now move in with the Laplace transform to get a formula for the solution of

$$y''[t] + 1.8\,y'[t] + 8.3\,y[t] = \text{approxf}[t]$$

with y[0] = 4.7 and y′[0] = −0.9.

Here you go:

```
Clear[f, t, approxY, approxy, s]
b = 1.8; c = 8.3;
f[t_] = approxf[t];
ystarter = 4.7; yprimestarter = 0.9;
approxY[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
     b ystarter + s ystarter + yprimestarter)
```

$$\frac{9.36 + \frac{3.37978}{s} + 4.7\,s - \frac{0.898218}{\pi^2+s^2} - \frac{2.35661\,s}{\pi^2+s^2} - \frac{0.492386}{4\pi^2+s^2} - \frac{0.736414\,s}{4\pi^2+s^2}}{8.3 + 1.8\,s + s^2}$$

This is the Laplace transform of the the approximate solution.

To get the formula for the approximate solution, approxy[t]. you calculate the inverse Laplace transform of approxY[s].

The formula for the solution approxy[t] is:

```
approxy[t_] = Chop[Expand[InverseLaplaceTransform[approxY[s], s, t]]]
```

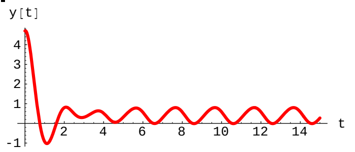
$$0.407202 + 4.11678\,E^{-0.9t}\,\mathrm{Cos}[2.73679\,t] + 0.154343\,\mathrm{Cos}[\pi t] + 0.0216786\,\mathrm{Cos}[2\pi t] + 2.12415\,E^{-0.9t}\,\mathrm{Sin}[2.73679\,t] - 0.3739\,\mathrm{Sin}[\pi t] - 0.00535028\,\mathrm{Sin}[2\pi t]$$

There it is!

A simple approximate formula for the solution.

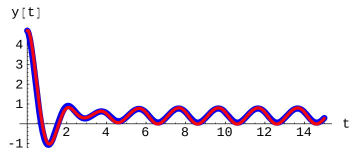
See it:

```
approxplot = Plot[approxy[t], {t, 0, endtime},
   PlotStyle -> {{Thickness[0.01], Red}}, PlotRange -> All,
   AxesLabel -> {"t", "y[t]"}];
```



Compare with what you got from `NDSolve`:

```
Show[ndsplot, approxplot];
```

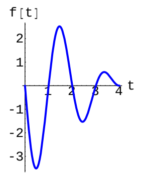


Both plots are fakes but they are both very good fakes.

B.5) Fourier integral fit of periodic functions

Here's how you can use integrals to go after a Fourier fit on an interval [0, L]:
Start with the function and its plot on [0, L]:

```
Clear[f, t]
f[t_] = Sin[π t] (t - 4);
L = 4;
Plot[f[t], {t, 0, L}, PlotStyle -> {{Thickness[0.02], Blue}},
  AxesLabel -> {"t", "f[t]"}];
```



To try to fit f[t] on [0, L], you fit with combinations of

$$E^{\frac{I k 2 \pi t}{L}}$$

this way:

```
Clear[A, t]
A[k_] := N[Integrate[f[t] E^(-I k (2 π) t/L), {t, 0, L}]/L];
Clear[k, complexfitter]
m = 5;
complexfitter[t_] = Chop[Sum[A[k] E^(I k (2 π) t/L), {k, -m, m}]]
```

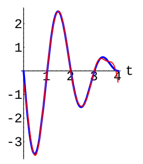
$$-0.31831 - 0.424413\,E^{-\frac{1}{2} I \pi t} - 0.424413\,E^{\frac{I \pi t}{2}} - (0.0795775 + 1.\,I)\,E^{-I \pi t} - (0.0795775 - 1.\,I)\,E^{I \pi t} + 0.254648\,E^{-\frac{3}{2} I \pi t} + 0.254648\,E^{\frac{3 I \pi t}{2}} + 0.106103\,E^{-2 I \pi t} + 0.106103\,E^{2 I \pi t} + 0.0606305\,E^{-\frac{5}{2} I \pi t} + 0.0606305\,E^{\frac{5 I \pi t}{2}}$$

```
Clear[realfitter, t]
realfitter[t_] = Chop[ComplexExpand[complexfitter[t]]]
```

$$-0.31831 - 0.848826\,\mathrm{Cos}\left[\frac{\pi t}{2}\right] - 0.159155\,\mathrm{Cos}[\pi t] + 0.509296\,\mathrm{Cos}\left[\frac{3\pi t}{2}\right] + 0.212207\,\mathrm{Cos}[2\pi t] + 0.121261\,\mathrm{Cos}\left[\frac{5\pi t}{2}\right] - 2.\,\mathrm{Sin}[\pi t]$$

Check out the fit with a plot:

```
Plot[{f[t], realfitter[t]}, {t, 0, L},
  PlotStyle -> {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  AxesLabel -> {"t", ""}];
```



The periodic nature of the Fourier fit ruins the fit at the ends, but inside [0, L], the fit is not all that bad.

□B.5.a)

How do you try to go after a better fit?

□ Answer:

The same way you go after a better fast Fourier fit: You increase m.

Try it:

```
Clear[A, t]
A[k_] := A[k] = NIntegrate[f[t] E^(-I k (2 π) t/L), {t, 0, L}]/L;
Clear[k, complexfitter]
m = 13;
complexfitter[t_] = Chop[Sum[A[k] E^(I k (2 π) t/L), {k, -m, m}]]
```

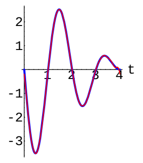
$$-0.31831-0.424413\,E^{-\frac{1}{2}I\pi t}-0.424413\,E^{\frac{I\pi t}{2}}-(0.0795775+1.\,I)\,E^{-I\pi t}-(0.0795775-1.\,I)\,E^{I\pi t}+0.254648\,E^{-\frac{3}{2}I\pi t}+0.254648\,E^{\frac{3I\pi t}{2}}+0.106103\,E^{-2I\pi t}+0.106103\,E^{2I\pi t}+0.0606305\,E^{-\frac{5}{2}I\pi t}+0.0606305\,E^{\frac{5I\pi t}{2}}+0.0397887\,E^{-3I\pi t}+0.0397887\,E^{3I\pi t}+0.0282942\,E^{-\frac{7}{2}I\pi t}+0.0282942\,E^{\frac{7I\pi t}{2}}+0.0212207\,E^{-4I\pi t}+0.0212207\,E^{4I\pi t}+0.0165356\,E^{-\frac{9}{2}I\pi t}+0.0165356\,E^{\frac{9I\pi t}{2}}+0.0132629\,E^{-5I\pi t}+0.0132629\,E^{5I\pi t}+0.0108824\,E^{-\frac{11}{2}I\pi t}+0.0108824\,E^{\frac{11I\pi t}{2}}+0.00909457\,E^{-6I\pi t}+0.00909457\,E^{6I\pi t}+0.0077166\,E^{-\frac{13}{2}I\pi t}+0.0077166\,E^{\frac{13I\pi t}{2}}$$

```
Clear[realfitter, t]
realfitter[t_] = Chop[ComplexExpand[complexfitter[t]]]
```

$$-0.31831-0.848826\,\text{Cos}\left[\frac{\pi t}{2}\right]-0.159155\,\text{Cos}[\pi t]+0.509296\,\text{Cos}\left[\frac{3\pi t}{2}\right]+0.212207\,\text{Cos}[2\pi t]+0.121261\,\text{Cos}\left[\frac{5\pi t}{2}\right]+0.0795775\,\text{Cos}[3\pi t]+0.0565884\,\text{Cos}\left[\frac{7\pi t}{2}\right]+0.0424413\,\text{Cos}[4\pi t]+0.0330712\,\text{Cos}\left[\frac{9\pi t}{2}\right]+0.0265258\,\text{Cos}[5\pi t]+0.0217648\,\text{Cos}\left[\frac{11\pi t}{2}\right]+0.0181891\,\text{Cos}[6\pi t]+0.0154332\,\text{Cos}\left[\frac{13\pi t}{2}\right]-2.\,\text{Sin}[\pi t]$$

Check out the fit with a plot:

```
Plot[{f[t], realfitter[t]}, {t, 0, L},
 PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
 AxesLabel → {"t", ""}];
```



That's almost as good as it can get. If f[0] had been the same as f[L], then the trouble near the endpoints would have disappeared.

□B.5.b.i) The formula

$$A[k] = \frac{1}{L}\int_0^L f[t]\,E^{-\frac{Ik(2\pi)t}{L}}\,dt$$

What's the idea behind the formula

$$A[k] = \frac{1}{L}\int_0^L f[t]\,E^{-\frac{Ik(2\pi)t}{L}}\,dt$$

as used above?

□**Answer:**

The idea:

You go for the whole ball of wax in one gulp.

Given a function f[t] with period L, you go ahead and assume that f[t] can be written in a form like this:

```
m = 3;
Clear[k, f, complexfitter, A, L]
complexfitter[t_] = Sum[A[k] E^(I k 2 π t/L), {k, -m, m}]
```

$$E^{-\frac{6I\pi t}{L}}A[-3]+E^{-\frac{4I\pi t}{L}}A[-2]+E^{-\frac{2I\pi t}{L}}A[-1]+A[0]+E^{\frac{2I\pi t}{L}}A[1]+E^{\frac{4I\pi t}{L}}A[2]+E^{\frac{6I\pi t}{L}}A[3]$$

Once you go with the assumption that f[t] = complexfitter[t], you gotta agree that

$$\int_0^L \text{complexfitter}[t]\,E^{-\frac{Ip(2\pi)t}{L}}\,dt \;=\; \int_0^L f[t]\,E^{-\frac{Ip(2\pi)t}{L}}\,dt,$$

no matter what p you go with.

Peek at what this tells you:

This might take a while.

```
Clear[p, f]
ColumnForm[Table[{Expand[∫_0^L complexfitter[t] E^(-I p (2 π) t/L) ⅆt],
   " must be", ∫_0^L f[t] E^(-I p (2 π) t/L) ⅆt},
  {p, -m, m}]]
```

$\{L\,A[-3],\ \text{must be},\ \int_0^L E^{\frac{6I\pi t}{L}} f[t]\,dt\}$

$\{L\,A[-2],\ \text{must be},\ \int_0^L E^{\frac{4I\pi t}{L}} f[t]\,dt\}$

$\{L\,A[-1],\ \text{must be},\ \int_0^L E^{\frac{2I\pi t}{L}} f[t]\,dt\}$

$\{L\,A[0],\ \text{must be},\ \int_0^L f[t]\,dt\}$

$\{L\,A[1],\ \text{must be},\ \int_0^L E^{-\frac{2I\pi t}{L}} f[t]\,dt\}$

$\{L\,A[2],\ \text{must be},\ \int_0^L E^{-\frac{4I\pi t}{L}} f[t]\,dt\}$

$\{L\,A[3],\ \text{must be},\ \int_0^L E^{-\frac{6I\pi t}{L}} f[t]\,dt\}$

Your eyes lead you to the formula

$$A[k] = \frac{1}{L}\int_0^L f[t]\,E^{-\frac{Ik(2\pi)t}{L}}\,dt.$$

This is the formula that was used in part i) so successfully.

Explanation over.

□B.5.b.ii)

Not so fast.
Wasn't this supposed to be a math course?
You know very well that a false assumption like

f[t] = complexfitter[t]

can be used to explain anything.
How do you justify this false assumption?

□**Answer:**

Good question.

The reason it works is that it is almost true.

Reason: For most functions f[t] with period L, only a bureaucratic bean-counter armed with a good magnifying glass could ever tell the difference between f[t] and the complexfitter you get with a very high m.

Try it on

$$f[t] = \sqrt{1 - \text{Cos}[\pi t]}$$

Because the period of Cos[t] is 2 π, the period of f[t] is L = 2.

```
Clear[f, t, x]
f[t_] = √(1 - Cos[π t]);
L = 2;
m = 6;
Clear[k, complexfitter, A]
A[k_] := N[1/L] NIntegrate[f[x] E^(-I k (2 π) x/L), {x, 0, L}, AccuracyGoal → 3];
complexfitter[t_] = Chop[Sum[A[k] E^(I k (2 π) t/L), {k, -m, m}]]
```

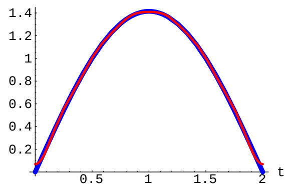
$$0.900316-0.300105\,E^{-I\pi t}-0.300105\,E^{I\pi t}-0.0600211\,E^{-2I\pi t}-0.0600211\,E^{2I\pi t}-0.0257233\,E^{-3I\pi t}-0.0257233\,E^{3I\pi t}-0.0142907\,E^{-4I\pi t}-0.0142907\,E^{4I\pi t}-0.0090941\,E^{-5I\pi t}-0.0090941\,E^{5I\pi t}-0.00629592\,E^{-6I\pi t}-0.00629592\,E^{6I\pi t}$$

This time the A[k]'s are calculated with NIntegrate.

```
Clear[realfitter, t]
realfitter[t_] = Chop[ComplexExpand[complexfitter[t]]]
```

$$0.900316-0.600211\,\text{Cos}[\pi t]-0.120042\,\text{Cos}[2\pi t]-0.0514466\,\text{Cos}[3\pi t]-0.0285815\,\text{Cos}[4\pi t]-0.0181882\,\text{Cos}[5\pi t]-0.0125918\,\text{Cos}[6\pi t]$$

```
cycles = 4;
Plot[{f[t], realfitter[t]}, {t, 0, L},
 PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
 AxesLabel → {"t", ""}, PlotRange → All];
```



Tell that miserable bean-counter to get out of your life.

□B.5.c.i) The connection between fast Fourier Fit and Fourier integral fit

Go with f[t] = t (3 – t) on [0, L] with L = 3 and compare what you get from Fourier integral fit and fast Fourier fit.

```
Clear[f, t]
f[t_] = t (3 - t);
L = 3;
Clear[A, t]
```

$$\texttt{A[k\_] := A[k] = N}\left[\frac{\int_0^L \texttt{f[t]}\, E^{-\frac{I k (2\pi) t}{L}}\, dt}{L}\right];$$

```
Clear[k, complexintegralfitter]
m = 5;
```

$$\texttt{complexintegralfitter[t\_] = Chop}\left[\sum_{k=-m}^{m} \texttt{A[k]}\, E^{\frac{I k (2\pi) t}{L}}\right]$$

$$1.5 - 0.455945\, E^{-\frac{2}{3} I \pi t} - 0.455945\, E^{\frac{2 I \pi t}{3}} - 0.113986\, E^{-\frac{4}{3} I \pi t} - 0.113986\, E^{\frac{4 I \pi t}{3}} - 0.0506606\, E^{-2 I \pi t} - 0.0506606\, E^{2 I \pi t} - 0.0284966\, E^{-\frac{8}{3} I \pi t} - 0.0284966\, E^{\frac{8 I \pi t}{3}} - 0.0182378\, E^{-\frac{10}{3} I \pi t} - 0.0182378\, E^{\frac{10 I \pi t}{3}}$$

```
n = m + 1;
FastFourierfit[f, L, n, t]
```

$$1.48958 - 0.466506\, E^{-\frac{2}{3} I \pi t} - 0.466506\, E^{\frac{2 I \pi t}{3}} - 0.125\, E^{-\frac{4}{3} I \pi t} - 0.125\, E^{\frac{4 I \pi t}{3}} - 0.0625\, E^{-2 I \pi t} - 0.0625\, E^{2 I \pi t} - 0.0416667\, E^{-\frac{8}{3} I \pi t} - 0.0416667\, E^{\frac{8 I \pi t}{3}} - 0.0334936\, E^{-\frac{10}{3} I \pi t} - 0.0334936\, E^{\frac{10 I \pi t}{3}}$$

They are almost the same.
Subtract them:

```
Expand[complexintegralfitter[t] - FastFourierfit[f, L, n, t]]
```

$$0.0104167 + 0.010561\, E^{-\frac{2}{3} I \pi t} + 0.010561\, E^{\frac{2 I \pi t}{3}} + 0.0110137\, E^{-\frac{4}{3} I \pi t} + 0.0110137\, E^{\frac{4 I \pi t}{3}} + 0.0118394\, E^{-2 I \pi t} + 0.0118394\, E^{2 I \pi t} + 0.0131701\, E^{-\frac{8}{3} I \pi t} + 0.0131701\, E^{\frac{8 I \pi t}{3}} + 0.0152558\, E^{-\frac{10}{3} I \pi t} + 0.0152558\, E^{\frac{10 I \pi t}{3}}$$

Look at those puny coefficients reflecting how close the fast Fourier fit is to the Fourier integral fit.
Try it again with a bigger m:

```
Clear[k, complexintegralfitter]
m = 10;
```

$$\texttt{complexintegralfitter[t\_] = Chop}\left[\sum_{k=-m}^{m} \texttt{A[k]}\, E^{\frac{I k (2\pi) t}{L}}\right];$$

```
n = m + 1;
Expand[complexintegralfitter[t] - FastFourierfit[f, L, n, t]]
```

$$0.00309917 + 0.00311185\, E^{-\frac{2}{3} I \pi t} + 0.00311185\, E^{\frac{2 I \pi t}{3}} + 0.00315039\, E^{-\frac{4}{3} I \pi t} + 0.00315039\, E^{\frac{4 I \pi t}{3}} + 0.00321633\, E^{-2 I \pi t} + 0.00321633\, E^{2 I \pi t} + 0.00331238\, E^{-\frac{8}{3} I \pi t} + 0.00331238\, E^{\frac{8 I \pi t}{3}} + 0.00344268\, E^{-\frac{10}{3} I \pi t} + 0.00344268\, E^{\frac{10 I \pi t}{3}} + 0.00361324\, E^{-4 I \pi t} + 0.00361324\, E^{4 I \pi t} + 0.00383251\, E^{-\frac{14}{3} I \pi t} + 0.00383251\, E^{\frac{14 I \pi t}{3}} + 0.00411247\, E^{-\frac{16}{3} I \pi t} + 0.00411247\, E^{\frac{16 I \pi t}{3}} + 0.00447017\, E^{-6 I \pi t} + 0.00447017\, E^{6 I \pi t} + 0.00493027\, E^{-\frac{20}{3} I \pi t} + 0.00493027\, E^{\frac{20 I \pi t}{3}}$$

How do you explain this striking similarity of the two Fourier fits?

□Answer:

The fast Fourier fit tries to fit at equally spaced points on the plot of f[t] on [0, L].

When you go with a high n in FastFourierfit[f, L, n, t], you are fitting so many densely packed points that FastFourierfit[f, L, n, t] is almost fitting all the points on the plot of f[t] on [0, L].

This nearly replicates the Fourier integral fit of f[t] on [0, L] because the Fourier integral fit tries to fit all the points on the plot of f[t] on [0, L].

□B.5.c.ii) Practicalities

Explain the following statement:
"Fast Fourier fit is almost always fast and practical; Fourier integral fit is sometimes slow and sometimes impractical and even impossible."

□Answer:

The Achilles heel of the Fourier integral fit is the integral

$$\frac{1}{L}\int_0^L f[t]\, E^{-\frac{I k (2\pi) t}{L}}\, dt$$

that must be done to calculate the coefficients. For some functions f[t], the only practical possibility is to go with `NIntegrate`. But the highly oscillatory nature of

$$f[t]\, E^{-\frac{I k (2\pi) t}{L}}$$

sometimes makes the integrals impractical even for for `NIntegrate`.

Fast Fourier fit carries none of this disappointing baggage. It runs quickly and accurately, provided no impulses are around.

□B.5.c.iii)

Why do folks want to do Fourier integral fit?

□Calculational Answer:

If the professors demand that the students work by hand, then the student has little choice between Fourier integral fit and fast Fourier fit.
Reason:
Doing fast Fourier fit by hand is completely out of the question.
Doing integrals by hand is not.

□Theoretical Answer:

In theoretical situations, most good folks want to use the Fourier integral fit because it gives them a specific formula to work with. But after the theory is developed and calculations begin, then the same folks usually go with fast Fourier fit.

□B.5.d.i) The advantage of fast Fourier fit over Fourier integral fit

```
 In their book, "Numerical Methods and Software" (Prentice-Hall, 1989),
            David Kahner, Cleve Moler and Stephen Nash say
       "Many knowledgeable people feel that [fast Fourier fit] is
        the single most important contribution to computing since
              the advent of the stored programming concept."
```

Fourier integral fit has been around for more than 200 years.
Fast Fourier fit has been around less than 50 years and really came into its own with the computer revolution of which you are a part.
Why are many compuer-literate folks so quick to de-emphasize Fourier integral fit in favor of fast Fourier fit?

□Answer:

The Achilles heel of the Fourier integral fit is the integral

$$\frac{1}{L}\int_0^L f[t]\, E^{-\frac{I k (2\pi) t}{L}}\, dt$$

which must be done to calculate the coefficients of the approximation. Even with *Mathematica*, these integrals can either be very slow or impossible. Numerical integration (via the NIntegrate instruction) is theoretically possible, but the highly oscillatory nature of

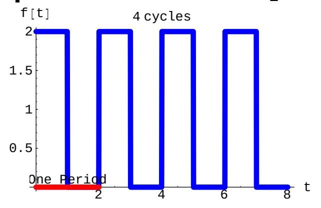
$$f[t]\, E^{-\frac{I k (2\pi) t}{L}}$$

sometimes makes the integrals impractical even for for numerical integration.

Fast Fourier fit does not care how the function is presented and it doesn't care about doing any integrals. Fast Fourier works rapidly and well for any function - other than impulse hits.
It is robust and fast.
Compare them on the square wave:

```
Clear[f, t]
f[t_] = Sign[Sin[π t]] + 1;
L = 2;
```

```
cycles = 4;
fplot = Plot[f[t], {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}}, AxesLabel → {"t", "f[t]"},
  PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



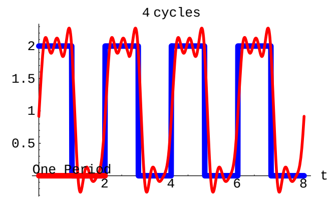
Go for a fast Fourier fit:

```
n = 6; Clear[fastapproxf]
fastapproxf[t_] = FastFourierfit[f, L, n, t]
```

```
1.08333 - (0.0833333 - 0.622008 I) E^(-I π t) - (0.0833333 + 0.622008 I) E^(I π t) +
 0.0833333 E^(-2 I π t) + 0.0833333 E^(2 I π t) - (0.0833333 - 0.166667 I) E^(-3 I π t) -
 (0.0833333 + 0.166667 I) E^(3 I π t) + 0.0833333 E^(-4 I π t) + 0.0833333 E^(4 I π t) -
 (0.0833333 - 0.0446582 I) E^(-5 I π t) - (0.0833333 + 0.0446582 I) E^(5 I π t)
```

Check it out:

```
Clear[realfit]
realfit[t_] = Chop[ComplexExpand[fastapproxf[t]]];
fastfitplot = Plot[{f[t], realfit[t]}, {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  AxesLabel → {"t", ""},
  PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



Now do the same thing with the integral Fourier fit:

Try numerical integration via **NIntegrate**.

```
Clear[A, t];
A[k_] := A[k] = N[NIntegrate[f[t] E^(-I k (2π) t / L), {t, 0, L}] / L];
Clear[k, integralfitter]
m = 5;
integralfitter[t_] = Chop[ Sum[A[k] E^(I k (2π) t / L), {k, -m, m}] ]
```

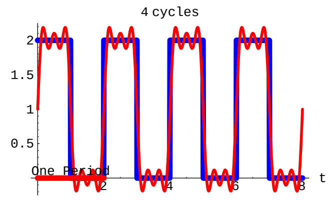
```
NIntegrate::ncvb : NIntegrate failed to converge to prescribed accuracy after 7
   recursive bisections in t near t = 2.58979720715110728`*^-12.
```

```
1. + 0.63662 I E^(-I π t) - 0.63662 I E^(I π t) + 0.212207 I E^(-3 I π t) -
 0.212207 I E^(3 I π t) + 0.127324 I E^(-5 I π t) - 0.127324 I E^(5 I π t)
```

This gives an answer, but *Mathematica* is telling you that it doesn't quite trust the answer.

Check it out:

```
Clear[realfit]
realfit[t_] = Chop[ComplexExpand[integralfitter[t]]];
integralfitplot = Plot[{f[t], realfit[t]}, {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  AxesLabel → {"t", ""},
  PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



The calculation of the integral fit took a lot longer than the calculation of the fast fit.

Fast Fourier fit went through this like a hot knife in butter.

Fourier integral fit went through this like a butter knife cutting steak.

□B.5.d..ii)

If you are working with hand calculations only, why is Fourier integral fit your only real choice?

□ **Answer:**

Fast Fourier fit involves solving many linear equations. Solving these equations by hand is just out of the question.

This leaves you no choice but Fourier integral fit. That's why you often see the Fourier integral fit in the old-fashioned courses that emphasize hand calculation.

DE.03 Laplace Transform and Fourier Analysis Tutorials

T.1) Using Fourier Fit to try to detect periodic forcing functions that force some undamped oscillators to beat or go into near or true resonance

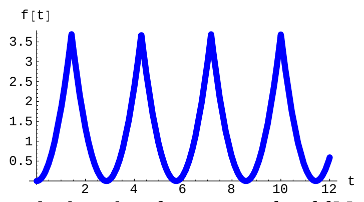
□T.1.a.i)

Activate this code.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];


Fourierfitters[L_, n_, t_] := Table[E^(2 π I k t / L),
     {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
 Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]


FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

Look at this periodic function f[t]:

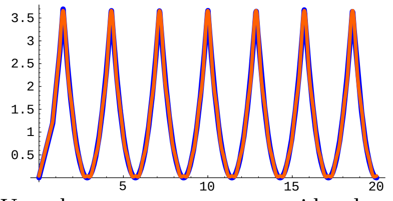
```
Clear[f, t];
f[t_] = 14.8 (0.35 t - Round[0.35 t])^2 ;
thigh = 12;
fplot =
 Plot[f[t], {t, 0, thigh}, PlotStyle → {{Thickness[0.02], Blue}},
  PlotRange → All, AxesLabel → {"t", "f[t]"}, AspectRatio → 1/2];
```



Now look at this fast Fourier fit of f[t]:

```
L = 1/0.35;
n = 7;
Clear[approxf];
approxf[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
fitplot = Plot[{f[t], approxf[t]},
   {t, 0, 20}, AspectRatio → 1/2, PlotRange → All, PlotStyle →
    {{Thickness[0.015], Blue}, {Thickness[0.01], CadmiumOrange}}];
```

```
1.24592 - 1.52498 Cos[2.19911 t] + 0.401106 Cos[4.39823 t] -
 0.194244 Cos[6.59734 t] + 0.123532 Cos[8.79646 t] -
 0.0930221 Cos[10.9956 t] + 0.0794439 Cos[13.1947 t]
```



Use what you see to come up with values of c so that the undamped forced oscillator

$$y''[t] + c\, y[t] = f[t]$$

are fried by going into resonance or near resonance (beating) .

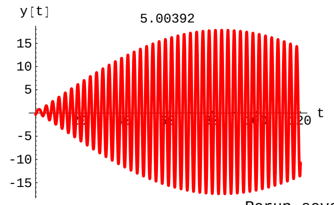
Answer:

Take another look at that good Fourier fit:

```
approxf[t]
```

```
1.24592 - 1.52498 Cos[2.19911 t] + 0.401106 Cos[4.39823 t] -
 0.194244 Cos[6.59734 t] + 0.123532 Cos[8.79646 t] -
 0.0930221 Cos[10.9956 t] + 0.0794439 Cos[13.1947 t]
```

```
f[t_] = approxf[t];
ystarter = Random[Real, {-1, 1}];
yprimestarter = Random[Real, {-1, 1}];
approxY[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter);
approxy[t_] =
 Chop[Expand[InverseLaplaceTransform[approxY[s], s, t]]];

approxplot = Plot[approxy[t], {t, 0, endtime},
  PlotStyle -> {{Thickness[0.01], Red}}, PlotPoints -> 2 endtime,
  AspectRatio -> 1 / GoldenRatio,
     PlotRange -> All, AxesLabel -> {"t", "y[t]"}, PlotLabel -> c];
```



Rerun several times.

To find even more resonating c's, look again at the good Fourier fit:

```
approxf[t]
```

```
1.24592 - 1.52498 Cos[2.19911 t] + 0.401106 Cos[4.39823 t] -
 0.194244 Cos[6.59734 t] + 0.123532 Cos[8.79646 t] -
 0.0930221 Cos[10.9956 t] + 0.0794439 Cos[13.1947 t]
```

That fairly big coefficient on the Cos[4.39823 t] term tells you that when you put c = 4.39823^2, you can count on the undamped forced oscillator

$$y''[t] + c\, y[t] = f[t]$$

to go into resonance or near resonance (beating) .

See it happen:

```
Clear[f, t, approxY, approxy, s];
b = 0;
c = 4.39823^2;
endtime = 80;
```

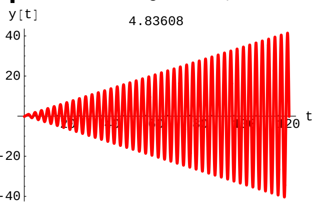
That big coefficient on the Cos[2.19911 t] term tells you that when you put c = 2.19911^2, you can count on the undamped forced oscillator

$$y''[t] + c\, y[t] = f[t]$$

to go into resonance or near resonance (beating) .

See it happen:

```
Clear[f, t, approxY, approxy, s];
b = 0;
c = 2.19911^2;
endtime = 120;
f[t_] = approxf[t];
ystarter = Random[Real, {-1, 1}];
yprimestarter = Random[Real, {-1, 1}];
approxY[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter);
approxy[t_] =
 Chop[Expand[InverseLaplaceTransform[approxY[s], s, t]]];

approxplot = Plot[approxy[t],
  {t, 0, endtime}, PlotStyle -> {{Thickness[0.01], Red}},
  PlotPoints -> 2 endtime, AspectRatio -> 1 / GoldenRatio,
     PlotRange -> All, AxesLabel -> {"t", "y[t]"}, PlotLabel -> c ];
```



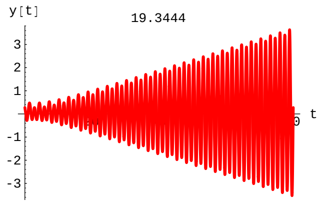
Fried.

In fact, if you take any c near 2.19911^2,

then you can still count on resonance or near resonance (beating):

```
Clear[f, t, approxY, approxy, s];
b = 0;
c = 2.19911^2 + Random[Real, {-0.3, 0.3}];
endtime = 120;
```

```
f[t_] = approxf[t];
ystarter = Random[Real, {-1, 1}];
yprimestarter = Random[Real, {-1, 1}];
approxY[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter);
approxy[t_] =
 Chop[Expand[InverseLaplaceTransform[approxY[s], s, t]]];

approxplot = Plot[approxy[t],
  {t, 0, endtime}, PlotStyle -> {{Thickness[0.01], Red}},
  PlotPoints -> 2 endtime, AspectRatio -> 1 / GoldenRatio,
     PlotRange -> All, AxesLabel -> {"t", "y[t]"}, PlotLabel -> c ];
```



To find even more resonating c's, look again at the good Fourier fit:

```
approxf[t]
```

```
1.24592 - 1.52498 Cos[2.19911 t] + 0.401106 Cos[4.39823 t] -
 0.194244 Cos[6.59734 t] + 0.123532 Cos[8.79646 t] -
 0.0930221 Cos[10.9956 t] + 0.0794439 Cos[13.1947 t]
```

That fairly big coefficient on the Cos[6.59734 t] term tells you that when you put c = 6.59734^2, you can count on the undamped forced oscillator

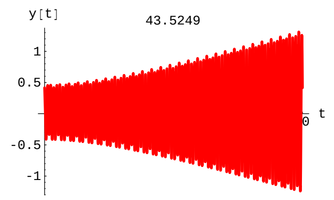
$$y''[t] + c\, y[t] = f[t]$$

to go into resonance or near resonance (beating) .

See it happen:

```
Clear[f, t, approxY, approxy, s]
b = 0;
c = 6.59734^2;
endtime = 80;
f[t_] = approxf[t];
ystarter = Random[Real, {-1, 1}];
```

```
yprimestarter = Random[Real, {-1, 1}];
approxY[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter);
approxy[t_] =
 Chop[Expand[InverseLaplaceTransform[approxY[s], s, t]]];

approxplot = Plot[approxy[t],
  {t, 0, endtime}, PlotStyle -> {{Thickness[0.01], Red}},
  PlotPoints -> 2 endtime, AspectRatio -> 1 / GoldenRatio,
     PlotRange -> All, AxesLabel -> {"t", "y[t]"}, PlotLabel -> c ];
```



You get the idea.

□T.1.a.ii)

Why did that work?

□Answer:

When you go with $y''[t] + c^2\, y[t] = 0$ (with $c > 0$).

the natural output consists of combinations of

Cos[c t] and Sin[c t].

When you force $y''[t] + c^2\, y[t] = 0$ with these combinations you get

$y''[t] + c^2\, y[t] =$ A Cos[c t] + B Sin[c t]

which is guaranteed to be in true resonance.

So when you see an approximate forcing function such as

2.0 Sin[2 t] − 0.4 Cos[4 t] + 0.5 Sin[6 t],

then you can be sure that each of the following undamped forceds

oscillators is in resonance or near resonance:

y"[t] + 4 y[t] = 2.0 Sin[2 t] − 0.4 Cos[4 t] + 0.5 Sin[6 t],

y"[t] + 16 y[t] = 2.0 Sin[2 t] − 0.4 Cos[4 t] + 0.5 Sin[6 t], and

y"[t] + 36 y[t] = 2.0 Sin[2 t] − 0.4 Cos[4 t] + 0.5 Sin[6 t].

T.2) The Laplace transform of yzeroinput[t] is the same as the product of Laplace transform of f[t] and the Laplace transform of yunitimpulse[t]

□T.2.a.i)

Given a forced oscillator diffeq

$$y''[t] + b\, y'[t] + c\, y[t] = f[t],$$

you know yunitimpulset[t] solves

$$y''[t] + b\, y'[t] + c\, y[t] = 0$$

with $y[0] = 0$ and $y'[0] = 1$.

The Laplace transform,Yunitimpulset[t] of yunitimpulset[t] is

```
Clear[f, t, b, c, Yunitimpulse , y, ystarter, yprimestarter, s]
f[t_] = 0;
ystarter = 0;

yprimestarter = 1;

Yunitimpulse[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter)
```

$\dfrac{1}{c + b\, s + s^2}$

Short and sweet.

Given a forced oscillator diffeq

$$y''[t] + b\, y'[t] + c\, y[t] = f[t],$$

you know yzeroinput[t] solves

$$y''[t] + b\, y'[t] + c\, y[t] = 0$$

with $y[0] = 0$ and $y'[0] = 1$.

The Laplace transform, Yzeroinput[t] of yzeroinput[t] is

```
Clear[f, t, b, c, Yzerinput, y, ystarter, yprimestarter, s]
ystarter = 0;
yprimestarter = 0;

Yzeroinput[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter)
```

$\dfrac{\text{LaplaceTransform}[f[t], t, s]}{c + b\, s + s^2}$

Two benefits:

→ This allows you to write down the Laplace transform of yzeroinput[t] directly in terms of the Laplace transform of f[t] and the coefficients b and c with no extra calculation.

→ And this reveals that the Laplace transform of yzeroinput[t] is the same as the product of Laplace transform of f[t] and the Laplace transform of yunitimpulse[t]:

```
LaplaceTransform[f[t], t, s] Yunitimpulse[s]
```

$\dfrac{\text{LaplaceTransform}[f[t], t, s]}{c + b\, s + s^2}$

Is the fact that the Laplace transform of yzeroinput[t] is the same as the product of Laplace transform of f[t] and the Laplace transform of yunitimpulse[t] an accident?

□Answer:

Get ready.

In Mathematics there are no accidents.

□T.2.a.ii)

Look at this calculation of the convolution of two cleared functions f[t] and g[t]:

```
Clear[f, g, h, t, s, x, y]

h[t_] = ∫_0^t g[t - x] f[x] ⅆx
```

$\int_0^t f[x]\, g[t - x]\, dx$

Calculate the Laplace transform of h[t]:

```
LaplaceTransform[h[t], t, s]
```

LaplaceTransform[f[t], t, s] LaplaceTransform[g[t], t, s]

This tells you that the Laplace transform of the convolution

$$h[t] = \int_0^t f[t - x]\, g[x]\, dx$$

is the Laplace transform of f[t] times the Laplace transform of g[t].

Use this fact to explain why the Laplace transform of yzeroinput[t] is the same as the product of Laplace transform of f[t] and the Laplace transform of yunitimpulse[t].

□Answer:

Remember that

$$\text{yzeroinput}[t] = \int_0^t \text{yunitimpulse}[t - x]\, f[x]\, dx.$$

This is the same as

$$h[t] = \int_0^t f[x]\, g[t - x]\, dx$$

with $g[t - x] = \text{yunitimpulse}[t - x]$ and $h[t] = \text{yzeroinput}[t]$.

According to what was said above,the Laplace transform of

yzeroinput[t] = h[t]

is the product of the Laplace transform of yunitimpulse[t] = g[t] and the Laplace transform of f[t].

T.3) A good way of screwing up your Fourier fits

□T.3.a)

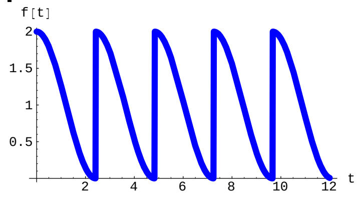
Activate this code.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, numtab, coeffs, t, L];
```

```
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];

Fourierfitters[L_, n_, t_] := Table[E^(2 π I k t/L),
    {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
 Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]

FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

Look at this plot of a periodic function:

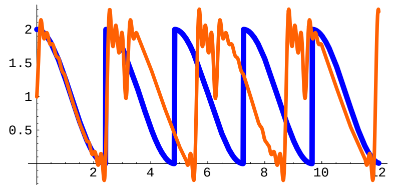
```
Clear[f, t];
f[t_] = Cos[1.3 t] Sign[Sin[1.3 t]] + 1 ;
thigh = 12;

fplot =
 Plot[f[t], {t, 0, thigh}, PlotStyle → {{Thickness[0.02], Blue}},
  PlotRange → All, AxesLabel → {"t", "f[t]"}, AspectRatio → 1/2];
```



Now look at this attempt at a fast Fourier fit of f[t]:

```
L = π;
n = 15;
Clear[approxf];

approxf[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
fitplot = Plot[{f[t], approxf[t]},
  {t, 0, thigh}, AspectRatio → 1/2, PlotRange → All, PlotStyle →
   {{Thickness[0.015], Blue}, {Thickness[0.01], CadmiumOrange}}];
```

```
1.14311 + 0.742814 Cos[2 t] + 0.0390951 Cos[4 t] - 0.257364 Cos[6 t] -
 0.188126 Cos[8 t] + 0.00806777 Cos[10 t] + 0.0504127 Cos[12 t] -
 0.0679726 Cos[14 t] - 0.145915 Cos[16 t] - 0.0810613 Cos[18 t] +
 0.0115204 Cos[20 t] - 0.00631859 Cos[22 t] - 0.0962819 Cos[24 t] -
 0.117143 Cos[26 t] - 0.0408109 Cos[28 t] + 0.415482 Sin[2 t] -
 0.25703 Sin[4 t] - 0.0860823 Sin[6 t] + 0.145233 Sin[8 t] +
 0.140967 Sin[10 t] - 0.0168037 Sin[12 t] - 0.0842575 Sin[14 t] +
 0.00337488 Sin[16 t] + 0.088743 Sin[18 t] + 0.0464058 Sin[20 t] -
 0.0482512 Sin[22 t] - 0.0522351 Sin[24 t] + 0.0307325 Sin[26 t] +
 0.0670608 Sin[28 t]
```

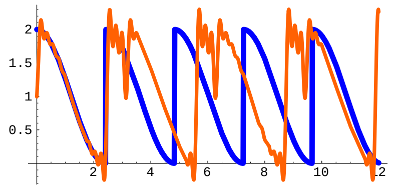


What went wrong?
How do you fix it?

□ **Answer:**

Take another look:

```
Show[fitplot];
```



This plot resulted from feeding L = π into the FourierFit instruction.

The trouble is that L = π is not a period of f[t].

The formula for f[t] is:

```
f[t]
```

```
1 + Cos[1.3 t] Sign[Sin[1.3 t]]
```

Calculate the period of f[t]:

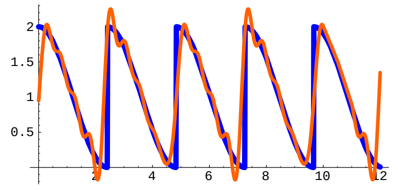
```
Solve[1.3 t == 2 π, t]
```

```
{{t → 4.83322}}
```

Set L = 4.83322 and run everything again:

```
L = 4.83322;
n = 10;
Clear[approxf]
approxf[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
fitplot = Plot[{f[t], approxf[t]},
  {t, 0, thigh}, AspectRatio → 1/2, PlotRange → All, PlotStyle →
   {{Thickness[0.015], Blue}, {Thickness[0.01], CadmiumOrange}}];
```

```
1.05 - 0.1 Cos[1.3 t] + 0.1 Cos[2.6 t] - 0.1 Cos[3.9 t] + 0.1 Cos[5.2 t] -
 0.1 Cos[6.5 t] + 0.1 Cos[7.8 t] - 0.1 Cos[9.1 t] + 0.1 Cos[10.4 t] -
 0.1 Cos[11.7 t] + 1.73205 × 10^-7 Sin[1.3 t] + 0.827636 Sin[2.6 t] +
 0.296261 Sin[5.2 t] + 0.150953 Sin[7.8 t] + 0.066791 Sin[10.4 t]
```



Lookin' real nice.

To get a good Fourier fit, you've got to set the period L of the Fourier fit to be equal to the period of the function you are fitting.

T.4) Inserting an extra t when the characteristic equation doesn't give two different solutions

□T.4.a.i)

Here's an oscillator differential equation with starter data:

```
b = 6;
c = 9;
Clear[t, y]
oscdiffeq = y''[t] + b y'[t] + c y[t] == 0
```

```
9 y[t] + 6 y'[t] + y''[t] == 0
```

Here's what happens when you use the charcteristic equation to go after formulas for the solution of this oscillator differential equation with sample starter data

y[0] = 1.2 and y′[0] = 3.4.

```
Clear[z]
charequation = z^2 + b z + c == 0;
zsols = Solve[charequation, z]
```

```
{{z → -3}, {z → -3}}
```

```
Clear[combo, C1, C2]
combo[t_] = C1 E^(zsols[[1,1,2]] t) + C2 E^(zsols[[2,1,2]] t)
```

```
C1 E^(-3 t) + C2 E^(-3 t)
```

```
ystarteq = combo[0] == 1.2
```

```
C1 + C2 == 1.2
```

```
yprimestarteq = combo'[0] == 3.4
```

```
-3 C1 - 3 C2 == 3.4
```

```
Csols = Solve[{ystarteq, yprimestarteq}, {C1, C2}]
```

```
RowReduce::luc : Result for RowReduce of badly conditioned matrix
   {{1., 1., -1.2}, {-3., -3., -3.4}} may contain significant numerical errors.
```

```
{{C1 → -4.20336 × 10^16, C2 → 4.20336 × 10^16}}
```

What went wrong?

□ **Answer:**

Look at combo[t]:

```
combo[t]
```

```
C1 E^(-3 t) + C2 E^(-3 t)
```

At first glance there appear to be two degrees of freedom here, but because the charactertistic equation has only one solution, you have only one degree of freedom. This is not enough to allow you to match up with the starter data on y[0] and y′[0].

□T.4.a.i) Inserting a t

What can you do to get around this problem?

□**Answer:**

Old DiffEq hands know how to tweak combo[t].

Here's how they do it:

Look at combo[t]:

```
combo[t]
```

$C1\,E^{-3t} + C2\,E^{-3t}$

Insert a t onto one of the terms:

```
Clear[bettercombo]
bettercombo[t_] = combo[t] /. C2 E^(-3 t) → t C2 E^(-3 t)
```

$C1\,E^{-3t} + C2\,E^{-3t}\,t$

Try it out:

```
ExpandAll[
oscdiffeq /. {y[t] -> bettercombo[t],
      y'[t] -> bettercombo'[t],
      y''[t] -> bettercombo''[t]}]
True
```

The upshot:

bettercombo[t] solves the differential equation and gives you the two genuine degrees of freedom you need to match up with the starter data on y[0] and y′[0].

```
ystarteq = bettercombo[0] == 1.2;
yprimestarteq = bettercombo′[0] == 3.4;
Csols = Solve[{ystarteq, yprimestarteq}, {C1, C2}]
{{C1 → 1.2, C2 → 7.}}
```

This gives you the exact formula:

```
Clear[yformula]
yformula[t] = bettercombo[t] /. Csols[[1]]
```

$1.2\,E^{-3t} + 7.\,E^{-3t}\,t$

Inserting the extra t did the trick.

□T.4.a.ii)

When do you insert an extra t as in part i) above?

□**Answer:**

If there are two different solutions of the characteristic equation, then you don't insert any extra t. In fact if you do, you will have a guaranteed screw-up.

If the characteristic equation has only one solution (double root), then you need to insert the t as in part i) above.

□T.4.a.iii) How important is the issue of the extra t?

How important is the issue of the extra t?

□**Answer:**

If you want an exact formula, it's important.

If you are willing to settle for a very good approximate formula then it's not an issue at all.

Watch this:

```
b = 3;
c = (b/2)^2;
Clear[t, y];
oscdiffeq = y″[t] + b y′[t] + c y[t] == 0
```

$\frac{9\,y[t]}{4} + 3\,y'[t] + y''[t] == 0$

Here's what happens when you use the characteristic equation to go after formulas for the solution of this oscillator differential equation with starter data

y[0] = 2 and y′[0] = 1.

```
Clear[z]
charequation = z^2 + b z + c == 0;
zsols = Solve[charequation, z]
```

$\{\{z \to -\frac{3}{2}\}, \{z \to -\frac{3}{2}\}\}$

Only one solution of the characteristic equation. If you want an exact formula, then you need to insert the t:

```
Clear[exacty]
exactsol = DSolve[{oscdiffeq, y[0] == 2, y′[0] == 1}, y[t], t];
exacty[t_] = y[t] /. exactsol[[1]]
```

$E^{-3t/2}\,(2 + 4\,t)$

There's that extra t.

If you are willing to settle for a good approximate formula, you look at the original differential equation:

```
b = 3;
c = (3/2)^2;
Clear[t, y];
diffeq = y″[t] + b y′[t] + c y[t] == 0
```

$\frac{9\,y[t]}{4} + 3\,y'[t] + y''[t] == 0$

Tweak the damping term b a little bit to get a nearby oscillator:

```
tweakedb = 1.0001 b;
Clear[t, y]
nearbydiffeq = y″[t] + tweakedb y′[t] + c y[t] == 0
```

$\frac{9\,y[t]}{4} + 3.0003\,y'[t] + y''[t] == 0$

Solve the nearby oscillator diffeq via the characteristic equation:

```
Clear[z]
charequation = z^2 + tweakedb z + c == 0;
zsols = Solve[charequation, z]
{{z → -1.52136}, {z → -1.47894}}
Clear[combo, C1, C2]
combo[t_] = C1 E^(zsols[[1,1,2]] t) + C2 E^(zsols[[2,1,2]] t)
```

$C1\,E^{-1.52136\,t} + C2\,E^{-1.47894\,t}$

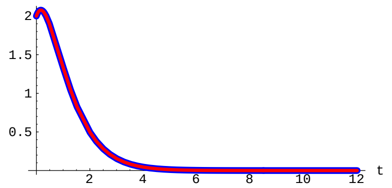
```
ystarteq = combo[0] == 2
C1 + C2 == 2
yprimestarteq = combo′[0] == 1
-1.52136 C1 - 1.47894 C2 == 1
Csols = Solve[{ystarteq, yprimestarteq}, {C1, C2}]
{{C1 → -93.2856, C2 → 95.2856}}
Clear[approxy]
approxy[t_] = combo[t] /. Csols[[1]]
```

$-93.2856\,E^{-1.52136\,t} + 95.2856\,E^{-1.47894\,t}$

No extra t.

Compare the two solutions:

```
endtime = 12;
Plot[{exacty[t], approxy[t]}, {t, 0, endtime},
  PlotStyle → {{Blue, Thickness[0.02]}, {Red, Thickness[0.01]}},
  PlotRange → All, AxesLabel → {"t", ""}, AspectRatio → 1/2];
```



Sharing ink all the way.

You can't tell the difference without a scorecard.

The upshot: The matter of inserting the t is important only when theoretical exactness is paramount.

In practical situations, you can almost always tweak the damping term to get good, practical results.

□T.4.b.i)

Use the Laplace transform to explain where the extra t comes from.

□**Answer:**

Here's another oscillator differential equation which needs the extra t:

```
b = 8;
c = 16;
Clear[t, y];
oscdiffeq = (y''[t] + b y'[t] + c y[t] == 0)
```

```
16 y[t] + 8 y′[t] + y″[t] == 0
```

Here's what happens when you use the characteristic equation to go after formulas for the solution of this oscillator differential equation with starter data

$y[0] = 2$ and $y'[0] = 3$.

```
Clear[z]
charequation = z^2 + b z + c == 0;
zsols = Solve[charequation, z]
```

```
{{z → -4}, {z → -4}}
```

Uh-oh.

Only one solution of the characteristic equation. To see where the extra t comes from, go to the Laplace Transform:

```
Clear[t, Y, approxy, s]
f[t_] = 0; ystarter = 2; yprimestarter = 3;
Y[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter);
y[t_] = Chop[Expand[InverseLaplaceTransform[Y[s], s, t]]]
```

$2 E^{-4t} + 11 E^{-4t}\, t$

Now you can see where the extra t comes from.

Look at the Laplace transform of the solution y[t]:

```
Apart[Y[s]]
```

$\frac{11}{(4+s)^2} + \frac{2}{4+s}$

The inverse Laplace transform of

$\frac{2}{s+4}$ is $2 E^{(-4t)}$

and the inverse Laplace transform of

$\frac{11}{(4+s)^2} = -11\, D[(\frac{1}{s+4}), s]$ is $11\, t\, E^{(-4t)}$:

```
2 InverseLaplaceTransform[1/(s + 4), s, t]
```

$2 E^{-4t}$

```
11 InverseLaplaceTransform[-D[1/(s + 4), s], s, t]
```

$11 E^{-4t}\, t$

That's where the t comes from.

□T.4.b.ii)

In the last part, knowing that the inverse Laplace transform of

$\frac{2}{s+4}$ is $2 E^{(-4t)}$,

you recognized that the inverse Laplace transform of

$\frac{11}{(4+s)^2} = -11\, D[\frac{1}{s+4}, s]$ is $-t\, E^{(-4t)}$.

(That's where the extra t really comes from.)

Is this a special case of a general rule?

□**Answer:**

You bet your sweet pocket calculator!

The rule is this:

If

Y[s] is the Laplace transform of y[t],

then the inverse Laplace transform of

$Y'[s]$ is $-t\, y[t]$.

It's even programmed into *Mathematica*:

```
Clear[s, t, y, Y]
Y[s_] = LaplaceTransform[y[t], t, s];
InverseLaplaceTransform[Y′[s], s, t]
```

```
-t y[t]
```

You can explain this sweet little rule as follows:

$$Y[s] = \int_0^\infty E^{-s t}\, y[t]\, dt$$
$$Y'[s] = \partial_s (\int_0^\infty E^{-s t}\, y[t]\, dt) = \int_0^\infty \partial_s (E^{-s t}\, y[t])\, dt$$
$$Y'[s] = \int_0^\infty E^{-s t}\, y[t]\, (-t)\, dt$$
$$Y'[s] = \int_0^\infty E^{-s t}\, (-t\, y[t])\, dt$$

This tells you that $Y'[s]$ is the Laplace transform of $-t\, y[t]$.

Nice, simple calculus.

□T.4.b.iii)

Is there a way you can see where the extra t comes from without resorting to the black box of the Laplace transform?

□**Answer:**

```
The idea for this explanation comes from V. I Arnold's book
                Ordinary Differential Equations,
Translated from the Russian by Roger Cooke, Springer-Verlag, 1992.
```

Yes there is!

The prototype oscillator differential equation whose characteristic equation has only one solution is:

```
Clear[oscdiffeq, r, y, t]
b = 2 r;
c = r^2;
oscdiffeq = y″[t] + b y′[t] + c y[t] == 0
```

```
r^2 y[t] + 2 r y′[t] + y″[t] == 0
```

Check:

```
Clear[z]
charequation = z^2 + b z + c == 0;
zsols = Solve[charequation, z]
```

```
{{z → -r}, {z → -r}}
```

Good.

Now tweak this oscillator this way:

```
Clear[nearbyoscdiffeq, y, h, t]
nearbyoscdiffeq[h_] = y″[t] + (2 r + h) y′[t] + r (r + h) y[t] == 0
```

```
r (h + r) y[t] + (h + 2 r) y′[t] + y″[t] == 0
```

When h is very small, then the two differential equations are almost the same.

```
ColumnForm[{oscdiffeq, nearbyoscdiffeq[0.00001]}]
```

```
r^2 y[t] + 2 r y′[t] + y″[t] == 0
r (0.00001 + r) y[t] + (0.00001 + 2 r) y′[t] + y″[t] == 0
```

Now look at formulas for solutions of both:

```
Clear[yexact, p, q]
exactsol = DSolve[{oscdiffeq, y[0] == p, y′[0] == q}, y[t], t];
yexact[t_] = y[t] /. exactsol⟦1⟧
```

$E^{-r t}\, (p + (q + p\, r)\, t)$

```
Clear[ynearby, p, q]
nearbysol =
  DSolve[{nearbyoscdiffeq[h], y[0] == p, y'[0] == q}, y[t], t];
ynearby[t_, h_] = y[t] /. nearbysol[[1]]
```

$E^{-r t}\left(-\frac{E^{(-h-r)\, t + r\, t}\, (q + p\, r)}{h} + \frac{h\, p + q + p\, r}{h}\right)$

Put:

```
Clear[g, x]
g[x_] = -E^(-x t)
```

$-E^{-t x}$

And notice that ynearby[t, h] is given by:

```
Clear[altynearby]
altynearby[t_, h_] = (q + p r) (g[r + h] - g[r]) / h + p / E^(r t)
```

$E^{-r t}\, p + \frac{(E^{-r t} - E^{-(h+r)\, t})\, (q + p\, r)}{h}$

Check:

```
Simplify[ynearby[t, h] - altynearby[t, h]]
```

```
0
```

Good.

As h closes in on 0,

nearbyoscdiffeq[h] closes in on oscdiffeq;

so

ynearby[t, h] closes in on yexact[t].

But

ynearby[t, h] = $\frac{(q+p\,r)\,(g[r+h]-g[r])}{h} + \frac{p}{E^{(r\,t)}}$,

and as h closes in on 0, this closes in on:

```
(q + p r) g′[r] + p/E^(r t)
```

$E^{-r\,t}\,p + E^{-r\,t}\,(q + p\,r)\,t$

This is the same as:

```
yexact[t]
```

$E^{-r\,t}\,(p + (q + p\,r)\,t)$

So the extra t comes from taking a derivative.

If this argument really appeals to you,
you may have a bright future as a mathematical scientist.

T.5) The Gibbs phenomenon

DiffEq&*Mathematica* is pleased to acknowledge that this problem is based in part on the exposition in Gilbert Strang's book "Introduction to Applied Mathematics" (Wellesley-Cambridge Press, 1986).

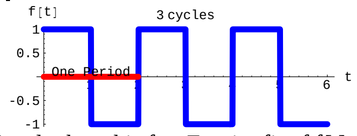
Activate this code.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];
```

```
Fourierfitters[L_, n_, t_] := Table[E^(2 π I k t/L),
     {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
  Part[InverseFourier[list], Drop[numtab[n], 1]]] /
   N[Sqrt[Length[list]]]

FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

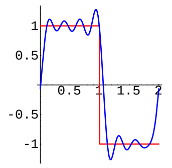
Look at this plot:

```
Clear[f, t];
f[t_] = Sign[Sin[π t]];
L = 2;
cycles = 3;

fplot =
 Plot[f[t], {t, 0, cycles L}, PlotStyle → {{Thickness[0.02], Blue}},
  AxesLabel → {"t", "f[t]"}, PlotLabel → "cycles" cycles,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



Now look at this fast Fourier fit of f[t] shown with f[t]:

```
n = 8;
Clear[realfit];
realfit[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];
fitplot = Plot[{f[t], realfit[t]}, {t, 0, L},
  PlotStyle → {{Thickness[0.01], Red}, {Blue}}];
```



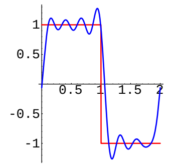
□T.5.a)

Late last century, the American mathematician Gibbs noticed theoretically (with no machine help) the same phenomenom you are seeing here. Lots of folks call this by the name "Gibbs phenomenon." What is the Gibbs phenomenon?

□**Answer:**

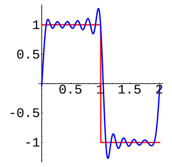
Look again:

```
Show[fitplot];
```



Increase the quality of the Fourier fit:

```
n = 12;
Clear[realfit];
realfit[t_] =
Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];

newfitplot = Plot[{f[t], realfit[t]}, {t, 0, L},
  PlotStyle -> {{Thickness[0.01], Red}, {Blue}}];
```



Look at the jump f[t] takes in the middle. Notice that the fast Fourier fit overshoots on one side of the jump with a compensating undershoot on the the other side of the jump.

Gibbs proved theoretically that when you go with a high quality fast Fourier fit (big n) of any periodic function f[t], then you will always get this phenomenon at a jump of f[t].

DE.03 Laplace Transform and Fourier Analysis
Give It a Try!

G.1) Using the Laplace transform to solve some forced exponential and forced oscillator diffeqs

□**G.1.a)**

Use the Laplace transform to come up with the exact formula for the solution of the forced exponential differential equation

$y'[t] - 0.5\,y[t] = 8\,\text{Cos}[2\,t]$

with $y[0] = 4$.

□**G.1.b)**

Use the Laplace transform to come up with the exact formula for the forced, damped oscillator coming from

$y''[t] + 0.7\,y'[t] + 2.8\,y[t] = 2\,\text{Sin}[\,3.2\,t]$

with $y[0] = 3.2$ and $y'[0] = -0.6$.

Inspect the formula to come up with a formula that describes the long-term behavior of this oscillator.

Give a nice plot.

□**G.1.c.i)**

Look at this:

```
Clear[f, F, s, t];
f[t_] = DiracDelta[t - 2.7];
F[s_] = LaplaceTransform[f[t], t, s]
```

$E^{-2.7 s}$

Use the integral definition of the Laplace transform to explain the output.

□G.1.c.ii)

Showcase the Laplace transform method and the Dirac delta function by coming up with the exact formula for the forced oscillator coming from

y″[t] + 0.5 y′[t] + 3.1 y[t] = f[t]

with f[t] = 10 DiracDelta[t – 8.5], y[0] = 8 and y′[0] = 2.

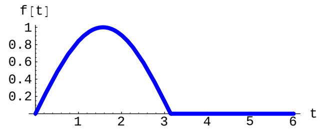
Show off your work with a good plot.

□G.1.d.i)

Look at this function:

```
Clear[f, t]
f[t_] = Sin[t] (1 - UnitStep[t - π])
Plot[f[t], {t, 0, 6}, PlotStyle → {{Thickness[0.015], Blue}},
 AspectRatio → 1/3, AxesLabel → {"t", "f[t]"}];
```

```
Sin[t] (1 - UnitStep[-π + t])
```



Explain why the formula

f[t] = Sin[t] (1 – UnitStep[t – π])

guarantees that

→ f[t] = Sin[t] for 0 ≤ t ≤ π and

→ f[t] = 0 for t > π.

□G.1.d.ii)

Stay with the same function f[t] = Sin[t] (1 – UnitStep[t – π]) as in part i) above.

Showcase the Laplace transform method to come with the exact formula for the forced oscillator coming from

y″[t] + 0.7 y′[t] + 9.1 y[t] = f[t]

with y[0] = 4 and y′[0] = 0.

Show off your work with a good plot.

□G.1.e.i)

Plot the new function f[t] described as follows:

→ f[t] = t for 0 ≤ t ≤ 2 and

→ f[t] = 0 for t > 2.

□G.1.e.ii)

Stay with the same function f[t] as in part i) above.

Use the Laplace transform method to come with the exact formula for the forced oscillator coming from

y″[t] + 0.7 y′[t] + 9.1 y[t] = f[t]

with y[0] = 4 and y′[0] = 0.

Show off your work with a good plot.

G.2) Periodic functions and some Fourier fits

Activate this code.

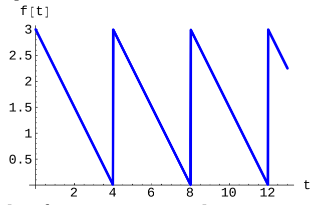
```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
  jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];

Fourierfitters[L_, n_, t_] := Table[E^(2 π I k t/L),
    {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
  Part[InverseFourier[list], Drop[numtab[n], 1]]] /
   N[Sqrt[Length[list]]]

FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

□G.2.a.i)

Look at this plot of

f[t] = 3 (Ceiling[$\frac{t}{4}$] – $\frac{t}{4}$):

```
Clear[f, t];
f[t_] = 3 (Ceiling[t/4] - t/4);
Plot[f[t], {t, 0, 13}, PlotStyle → {{Thickness[0.01], Blue}},
 AspectRatio → 1/GoldenRatio, AxesLabel → {"t", "f[t]"},
 PlotRange → All];
```



This function is periodic.

Identify the period L of f[t] and plot three cyles of f[t].

□G.2.a.ii)

Give a fairly decent, but not necessarily great, fast Fourier fit of this function on [0, L] and then plot three cyles of f[t] and your fast Fourier fit.

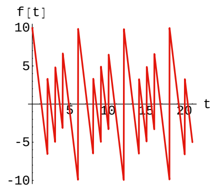
□G.2.b.i)

Look at this plot of

f[t] = 10 (Ceiling[$\frac{t}{3}$] + Floor[$\frac{t}{2}$] – $\frac{5 t}{6}$):

```
Clear[f, t];
f[t_] = 10 (Ceiling[t/3] + Floor[t/2] - 5 t/6);
```

```
Plot[
 f[t], {t, 0, 21}, PlotStyle → {{Thickness[0.01], DeepCadmiumRed}},
 AxesLabel → {"t", "f[t]"}];
```



This function is periodic.

Identify the period L of f[t] and plot three cyles of f[t].

□G.2.b.ii)

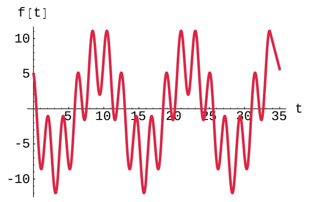
Give a fairly decent, but not necessarily great, fast Fourier fit of this function on [0, L] and then plot three cyles of f[t] and your fast Fourier fit.

□G.2.c)

Look at this plot of

f[t] = 5 Cos[3 t] – 7 Sin[$\frac{t}{2}$]:

```
Clear[f, t];
f[t_] = 5 Cos[3 t] - 7 Sin[t/2];
Plot[f[t], {t, 0, 35},
 PlotStyle → {{Thickness[0.01], PermanentRedViolet}},
 AxesLabel → {"t", "f[t]"}];
```



This function is periodic.
Identify the period L of f[t] as a multiple of π and plot two cyles of f[t].

G.3) Using Fourier fit and the Laplace transform to come up with good approximate formulas for some periodically forced oscillators

Activate this code.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];


Fourierfitters[L_, n_, t_] := Table[E^(2 π I k t/L),
     {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
 Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]

FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

☐G.3.a)

When you go after a reasonably accurate formula for the forced, damped oscillator coming from

$$y''[t] + 1.3\, y'[t] + 5.4\, y[t] = f[t]$$

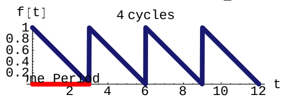
with $y[0] = 2.2$ and $y'[0] = -3.5$

and

with $f[t] = \text{Ceiling}[\frac{t}{3}] - \frac{t}{3}$,

you first look at the plot of f[t]:

```
Clear[f, t];
f[t_] = Ceiling[t/3] - t/3;
cycles = 4;
L = 3;
fplot = Plot[f[t], {t, 0, cycles L}, PlotStyle →
    {{Thickness[0.02], MidnightBlue}}, AxesLabel → {"t", "f[t]"},
  PlotLabel → "cycles" cycles, AspectRatio → 1/4,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



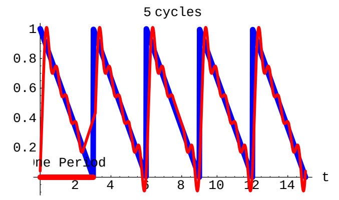
This f[t] is periodic with period L = 3.
Next you go after a reasonably good fast Fourier fit of this function and check it out:

```
L = 3;
Clear[approxf];
n = 6;
approxf[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
```

```
0.458333 - 0.0833333 Cos[2 π t/3] - 0.0833333 Cos[4 π t/3] -
 0.0833333 Cos[2 π t] - 0.0833333 Cos[8 π t/3] - 0.0833333 Cos[10 π t/3] +
 0.311004 Sin[2 π t/3] + 0.144338 Sin[4 π t/3] + 0.0833333 Sin[2 π t] +
 0.0481125 Sin[8 π t/3] + 0.0223291 Sin[10 π t/3]
```

Check it out:

```
cycles = 5;
fitplot = Plot[{f[t], approxf[t]}, {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  AxesLabel → {"t", ""},
  PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



Folks like to call those little overshoots and undershoots at the jumps by the name "Gibbs Phenomena."
For a word on this, see the Tutorials.

Looks pretty good.
Now you take over and use the Laplace transform to come up with a pretty decent approximate solution of

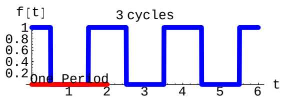
$$y''[t] + 1.3\, y'[t] + 5.4\, y[t] = f[t]$$

with $y[0] = 2.2$ and $y'[0] = -3.5$.

☐G.3.b.i)

Here is the plot of a periodic function:

```
Clear[f, t];
f[t_] = 0.5 (Sign[Cos[π t]] + 1);
L = 2;
cycles = 3;
fplot = Plot[f[t], {t, 0, cycles L},
  PlotStyle → {{Thickness[0.02], Blue}}, AxesLabel → {"t", "f[t]"},
  PlotLabel → "cycles" cycles, AspectRatio → 1/4,
  Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
    {Text["One Period", {L/2, 0.1}]}}];
```



This f[t] is periodic with period L = 2. Use fast Fourier fit to do a reasonably good job of fitting this function with combinations sine and cosine waves.
Don't be too picky.

☐G.3.b.ii)

Go with the same f[t] used in part i) and use the formula for the fast Fourier fit you got in part i) together with the Laplace transform to slam out an approximate formula for the solution of the forced oscillator

$$y''[t] + 0.2\, y'[t] + 3.7\, y[t] = f[t]$$

with $y[0] = 1$ and $y'[0] = 2$.

Plot your approximate formula.

☐G.3.b.iii)

Look at the formula you got in part ii) above.
What part of the formula reflects the starting data

$y[0] = 1$ and $y'[0] = 2$?

What part of the formula reflects the steady state behavior of the forced oscillator?

☐G.3.c)

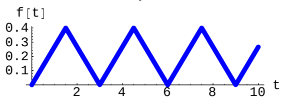
Use Fast Fourier fit and the Laplace transform to come up with a good approximate formula of the steady state behavior of the forced, damped oscillator coming from

$$y''[t] + 2.3\, y'[t] + 6.7\, y[t] = f[t]$$

with $y[0] = 4.3$ and $y'[0] = -2.5$

and with f[t] as specified below:

```
Clear[f, t];
f[t_] = 0.8 Abs[t/3 - Round[t/3]];
fplot = Plot[f[t], {t, 0, 10}, PlotStyle → {{Thickness[0.02], Blue}},
  AspectRatio → 1/4, AxesLabel → {"t", "f[t]"}];
```



This time you have to be careful about setting the period L.

G.4) Fourier analysis

Activate this code before you start.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];


Fourierfitters[L_, n_, t_] := Table[E^((2 π I k t)/L),
    {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
 Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]

FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

□G.4.a.i)

Here's a plot of the forced undamped oscillator

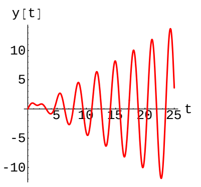
$y''[t] + 4\,y[t] = f[t]$

with $y[0] = 0$ and $y'[0] = 2$

and

with $f[t] = 5\,\mathrm{Sin}[t]^6$:

```
endtime = 25; Clear[f, y, t];
f[t_] = 5 Sin[t]^6; ndssol = NDSolve[
  {y''[t] + 4 y[t] == f[t], y[0] == 0, y'[0] == 2}, y[t], {t, 0, endtime}];
ndsy[t_] = y[t] /. ndssol[[1]];
ndsplot = Plot[ndsy[t], {t, 0, endtime},
  PlotStyle → {{Red, Thickness[0.01]}}, AxesLabel → {"t", "y[t]"}];
```



This is showing signs of resonance. When you showed this plot to that dork Calculus Cal, he said, "Look at the characteristic equation for the unforced oscillator

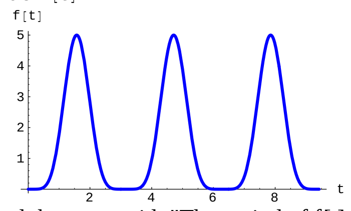
$y''[t] + 4\,y[t] = 0$:"

```
Clear[z]
Solve[z^2 + 4 == 0]
{{z → -2 I}, {z → 2 I}}
```

Cal then continued, "There must be some mistake because the only way you can put this oscillator in resonance is to force it with multiples of Cos[2 t] or Sin[2 t]."

You said, "Maybe that's the impression you got from the Tutorials, but I don't agree; look at f[t] and a plot:"

```
f[t]
Plot[f[t], {t, 0, 3 Pi},
 PlotStyle -> {{Thickness[0.01], Blue}},
 AxesLabel -> {"t", "f[t]"}];
5 Sin[t]^6
```



And then you said: "The period of f[t] is L = π ; look at a reasonably good Fourier fit of f[t]:"

```
Chop[ComplexExpand[FastFourierfit[f, π, 9, t]]]
1.5625 - 2.34375 Cos[2 t] + 0.9375 Cos[4 t] - 0.15625 Cos[6 t]
```

And then you said, "Now I see where the resonance came from." What did you mean?

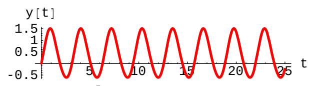
□G.4.a.ii)

At this point, Cal was totally clueless, and you decided to rub it in a bit by saying, "I think I can eliminate the resonance altogther f[t] by adding

2.34375 Cos[2 t]

to f[t]. Just take a look:"

```
Clear[y, t]
ndssol = NDSolve[{y''[t] + 4 y[t] == f[t] + 2.34375 Cos[2 t],
               y[0] == 0, y'[0] == 2}, y[t], {t, 0, endtime}];
y[t_] = y[t] /. ndssol[[1]];
adjustedplot =
 Plot[y[t], {t, 0, endtime}, PlotStyle → {{Red, Thickness[0.01]}},
  PlotRange → All, AxesLabel → {"t", "y[t]"}, AspectRatio → 1/5];
```



No resonance here.

How did you come up with this idea of filtering out the bad term?

□G.4.b.i)

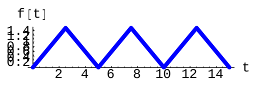
Later in your life, you are working for Ameritech. One fine morning your boss comes into your office saying , "There is a problem with an oscillator designed in the Columbus office. The oscillator comes from

$y''[t] + (1.2)^2\,y[t] = f[t]$,

with $y[0] = 0$ and $y'[0] = 0$

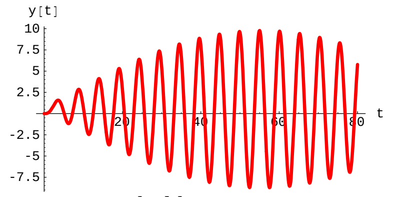
and with f[t] specified as follows:"

```
Clear[f, t];
f[t_] = 3 Abs[0.2 t - Round[0.2 t]];
fplot = Plot[f[t], {t, 0, 15}, PlotStyle → {{Thickness[0.02], Blue}},
  AxesLabel → {"t", "f[t]"}, AspectRatio → 1/5];
```



Your boss says, "Look at this plot to get an idea of what the trouble is:"

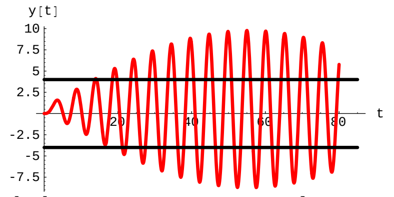
```
endtime = 80;
Clear[y, t, Derivative];
ndssol = NDSolve[{y''[t] + 1.2^2 y[t] == f[t], y[0] == 0, y'[0] == 0},
  y[t], {t, 0, endtime}, MaxSteps → 1500];
y[t_] = y[t] /. ndssol[[1]];
ndsplot =
 Plot[y[t], {t, 0, endtime}, PlotStyle → {{Red, Thickness[0.01]}},
  AspectRatio → 1/2, AxesLabel → {"t", "y[t]"}];
```



You say, "Looks like near resonance to me."

And then your boss says, "Just what I was afraid of. We can't tolerate an amplitude of more than 4 and this oscillator has amplitudes exceeding 4."

```
Show[ndsplot,
 Graphics[{Thickness[0.01], Line[{{0, 4}, {endtime + 5, 4}}]}],
 Graphics[{Thickness[0.01], Line[{{0, -4}, {endtime + 5, -4}}]}]];
```



The boss says: "We can't put a damper on this oscillator. Can you fix this by adding an extra term of the form

A Cos[B t] or A Sin[B t] with $-0.5 \le A \le 0.5$

to f[t] ?"

Remembering that little session with Cal in the Mathematica lab back at school, and noticing that the period of f[t] is L = 5, you look at:

```
N[Chop[ComplexExpand[FastFourierfit[f, 5, 12, t]]]]
```

```
0.75 - 0.611411 Cos[1.25664 t] - 0.0711294 Cos[3.76991 t] -
 0.0281083 Cos[6.28319 t] - 0.0165499 Cos[8.79646 t] -
 0.0122039 Cos[11.3097 t] - 0.0105972 Cos[13.823 t]
```

And then you say: "I'm not sure, but I'll give it my best shot."
Do it.

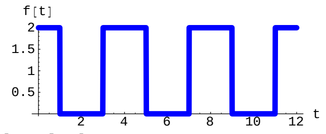
□G.4.b.ii)

This morning, your boss comes into your office saying, "We have another problem oscillator. It comes from

$$y''[t] + (1.6)^2\, y[t] = f[t]\ ,$$
$$\text{with } y[0] = 0 \text{ and } y'[0] = 0$$

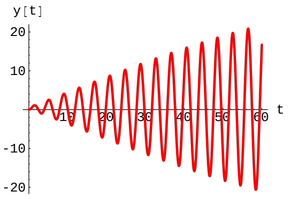
and with f[t] specified as follows:"

```
Clear[f, t];
f[t_] = Sign[Cos[0.5 π t]] + 1;
fplot = Plot[f[t], {t, 0, 12}, PlotStyle → {{Thickness[0.02], Blue}},
  AxesLabel → {"t", "f[t]"}, AspectRatio → 1/3];
```



"Take a look:"

```
endtime = 60;
Clear[y, t, Derivative];
ndssol = NDSolve[{y''[t] + 1.6^2 y[t] == f[t], y[0] == 0, y'[0] == 0},
  y[t], {t, 0, endtime}, MaxSteps → 2000];
y[t_] = y[t] /. ndssol[[1]];
Plot[y[t], {t, 0, endtime}, PlotStyle → {{Red, Thickness[0.01]}},
 AxesLabel → {"t", "y[t]"}];
```



And then your boss said, "We can't tolerate an amplitude of more than 2.0 as t runs from 0 to 60. Try to work your magic again to fix this this by adding an extra term of the form

$$A \cos[B\,t] \text{ or } A \sin[B\,t] \quad \text{with } -1.5 \le A \le 1.5$$

to f[t]."
What do you say and what do you do?

G.5) Mistakes, experiments and opinions

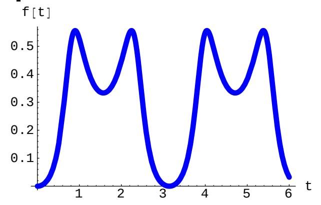
Activate this code.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];


Fourierfitters[L_, n_, t_] := Table[E^(2 π I k t/L),
     {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
 Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]

FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

□G.5.a)

That nitwit Calculus Cal starts to plot this periodic function:

```
Clear[f, t];
f[t_] = Sin[t]^2/(2 + Cos[4 t]);
cycles = 4;
Plot[f[t], {t, 0, 6}, PlotStyle → {{Thickness[0.02], Blue}},
 AxesLabel → {"t", "f[t]"}, AspectRatio → 1/GoldenRatio];
```



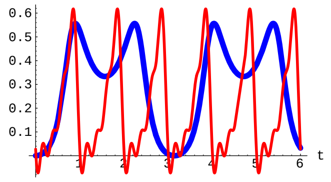
Intent on getting a decent fast Fourier fit with as little thought as possible, Cal copies and pastes some code:

```
L = 1; n = 5; Clear[t, realfastfit]
realfastfit[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]
```

```
0.199184 + 0.0143339 Cos[2 π t] - 0.0709849 Cos[4 π t] -
 0.0592728 Cos[6 π t] - 0.0556396 Cos[8 π t] - 0.24815 Sin[2 π t] -
 0.0886425 Sin[4 π t] - 0.0380883 Sin[6 π t] - 0.0170081 Sin[8 π t]
```

Smiling in satisfaction, Cal plots f[t] and his fast Fourier fit:

```
Plot[{f[t], realfastfit[t]}, {t, 0, 6},
 PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
 AxesLabel → {"t", ""}, AspectRatio → 1/GoldenRatio];
```

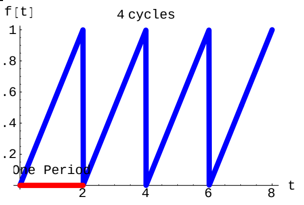


Cal is dumbfounded (as always).
Step in and tell Cal what he did wrong and then fix it so it looks as good as it can.

□G.5.b)

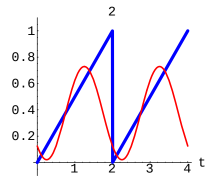
Here's a new periodic function:

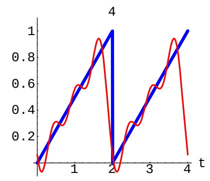
```
Clear[f, t];
f[t_] = t/2 - Floor[t/2];
L = 2;
cycles = 4;
Plot[f[t], {t, 0, cycles L},
 PlotStyle → {{Thickness[0.02], Blue}}, AxesLabel → {"t", "f[t]"},
 PlotLabel → "cycles" cycles, AspectRatio → 1/GoldenRatio,
 Epilog → {{Red, Thickness[0.02], Line[{{0, 0}, {L, 0}}]},
   {Text["One Period", {L/2, 0.1}]}}];
```

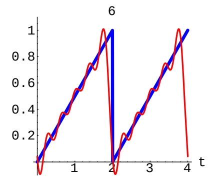


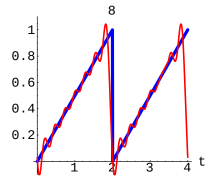
Now look at these plots which compare the plot of f[t] to plots of FastFourierFit[f, L, n, t]:

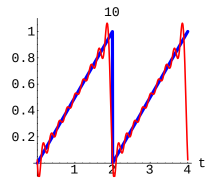
```
Clear[t, n, realfastfitter, comparisonplot];
realfastfitter[t_, n_] :=
 Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];
comparisonplot[n_] := Plot[{f[t], realfastfitter[t, n]}, {t, 0, 2 L},
  PlotStyle → {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  PlotRange → {-0.1, 1.1},
  AspectRatio → 1, AxesLabel → {"t", ""}, PlotLabel → n];
Table[comparisonplot[n], {n, 2, 10, 2}];
```











Grab and animate at various speeds.

What do these plots depict?
Does the Gibbs phenomenon show up?

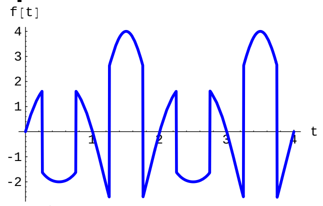
□G.5.c.i) Fast Fourier versus Fourier integral

What is the inherent advantage of using fast Fourier fit instead of the Fourier integral fit?

□G.5.c.ii)

Here's a periodic function:

```
Clear[f, t];
f[t_] = Sin[π t] (3 - Sin[π t]) Sign[Cos[2 π t]];
Plot[f[t], {t, 0, 4}, AspectRatio → 1/GoldenRatio,
 PlotStyle → {{Thickness[0.01], Blue}}, AxesLabel → {"t", "f[t]"}];
```



This function is periodic with period L = 2.
Here is a Fourier integral fit:

```
L = 2;
Clear[A, integralfitter,
 realintegralfitter, realfastfitter, fastfitter, k, t];
A[k_] := NIntegrate[f[t] E^(-I k (2 π) t / L), {t, 0, L}] / L;
Clear[k, complexintegralfitter];
m = 3;
integralfitter[t_] = Chop[Sum[A[k] E^(I k (2 π) t / L), {k, -m, m}]]
```

$0.31831 - 0.95493\, I\, E^{-I \pi t} + 0.95493\, I\, E^{I \pi t} - 0.31831\, E^{-2 I \pi t} - 0.31831\, E^{2 I \pi t} + 0.95493\, I\, E^{-3 I \pi t} - 0.95493\, I\, E^{3 I \pi t}$

Convert it to sines and cosines:

```
realintegralfitter[t_] = Chop[ComplexExpand[integralfitter[t]]]
```

$0.31831 - 0.63662\, \mathrm{Cos}[2 \pi t] - 1.90986\, \mathrm{Sin}[\pi t] + 1.90986\, \mathrm{Sin}[3 \pi t]$

And the corresponding fast Fourier fit:

```
n = m + 1;
fastfitter[t_] = FastFourierfit[f, L, n, t]
```

$0.25 + (0.75 - 0.75\, I)\, E^{-I \pi t} + (0.75 + 0.75\, I)\, E^{I \pi t} - (0.25 + 0.25\, I)\, E^{-2 I \pi t} - (0.25 - 0.25\, I)\, E^{2 I \pi t} - (0.75 - 0.75\, I)\, E^{-3 I \pi t} - (0.75 + 0.75\, I)\, E^{3 I \pi t}$

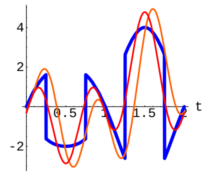
Convert it to sines and cosines:

```
realfastfitter[t_] = Chop[ComplexExpand[fastfitter[t]]]
```

$0.25 + 1.5\, \mathrm{Cos}[\pi t] - 0.5\, \mathrm{Cos}[2 \pi t] - 1.5\, \mathrm{Cos}[3 \pi t] - 1.5\, \mathrm{Sin}[\pi t] - 0.5\, \mathrm{Sin}[2 \pi t] + 1.5\, \mathrm{Sin}[3 \pi t]$

See both along with f[t]:

```
Plot[{f[t], realfastfitter[t], realintegralfitter[t]}, {t, 0, L},
PlotStyle -> {{Thickness[0.02], Blue},
   {Thickness[0.01], CadmiumOrange}, {Thickness[0.01], Red}},
 AspectRatio -> 1,
 AxesLabel -> {"t", ""}];
```



The plot of f[t] is blue.
The plot of the fast Fourier fit is orange.
The plot of the Fourier integral fit is red.

Raise m = n + 1 and do everything again:

```
m = 8;
L = 2;
Clear[A, integralfitter,
 realintegralfitter, realfastfitter, fastfitter, k, t];
A[k_] := NIntegrate[f[t] E^(-I k (2 π) t / L), {t, 0, L}] / L;
Clear[k, complexintegralfitter];
integralfitter[t_] = Chop[Sum[A[k] E^(I k (2 π) t / L), {k, -m, m}]]
```

$0.31831 - 0.95493\, I\, E^{-I \pi t} + 0.95493\, I\, E^{I \pi t} - 0.31831\, E^{-2 I \pi t} - 0.31831\, E^{2 I \pi t} + 0.95493\, I\, E^{-3 I \pi t} - 0.95493\, I\, E^{3 I \pi t} + 0.106103\, E^{-4 I \pi t} + 0.106103\, E^{4 I \pi t} + 0.31831\, I\, E^{-5 I \pi t} - 0.31831\, I\, E^{5 I \pi t} + 0.106103\, E^{-6 I \pi t} + 0.106103\, E^{6 I \pi t} - 0.31831\, I\, E^{-7 I \pi t} + 0.31831\, I\, E^{7 I \pi t} - 0.0212207\, E^{-8 I \pi t} - 0.0212207\, E^{8 I \pi t}$

Convert it to sines and cosines:

```
realintegralfitter[t_] = Chop[ComplexExpand[integralfitter[t]]]
```

$0.31831 - 0.63662\, \mathrm{Cos}[2 \pi t] + 0.212207\, \mathrm{Cos}[4 \pi t] + 0.212207\, \mathrm{Cos}[6 \pi t] - 0.0424413\, \mathrm{Cos}[8 \pi t] - 1.90986\, \mathrm{Sin}[\pi t] + 1.90986\, \mathrm{Sin}[3 \pi t] + 0.63662\, \mathrm{Sin}[5 \pi t] - 0.63662\, \mathrm{Sin}[7 \pi t]$

And the corresponding fast Fourier fit:

```
n = m + 1;
fastfitter[t_] = FastFourierfit[f, L, n, t]
```

$0.264376 - 0.793128\, I\, E^{-I \pi t} + 0.793128\, I\, E^{I \pi t} - 0.321714\, E^{-2 I \pi t} - 0.321714\, E^{2 I \pi t} + 1.13716\, I\, E^{-3 I \pi t} - 1.13716\, I\, E^{3 I \pi t} + 0.163531\, E^{-4 I \pi t} + 0.163531\, E^{4 I \pi t} + 0.15597\, I\, E^{-5 I \pi t} - 0.15597\, I\, E^{5 I \pi t} + 0.117812\, E^{-6 I \pi t} + 0.117812\, E^{6 I \pi t} - 0.550901\, I\, E^{-7 I \pi t} + 0.550901\, I\, E^{7 I \pi t} - 0.0918169\, E^{-8 I \pi t} - 0.0918169\, E^{8 I \pi t}$

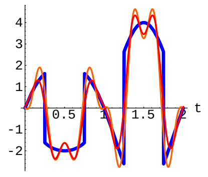
Convert it to sines and cosines:

```
realfastfitter[t_] = Chop[ComplexExpand[fastfitter[t]]]
```

$0.264376 - 0.643429\, \mathrm{Cos}[2 \pi t] + 0.327063\, \mathrm{Cos}[4 \pi t] + 0.235624\, \mathrm{Cos}[6 \pi t] - 0.183634\, \mathrm{Cos}[8 \pi t] - 1.58626\, \mathrm{Sin}[\pi t] + 2.27432\, \mathrm{Sin}[3 \pi t] + 0.311941\, \mathrm{Sin}[5 \pi t] - 1.1018\, \mathrm{Sin}[7 \pi t]$

See both along with f[t]:

```
Plot[{f[t], realfastfitter[t], realintegralfitter[t]},
 {t, 0, L}, PlotStyle → {{Thickness[0.02], Blue},
   {Thickness[0.01], CadmiumOrange}, {Thickness[0.01], Red}},
 AspectRatio → 1, AxesLabel → {"t", ""}];
```



The plot of f[t] is blue.
The plot of the fast Fourier fit is orange.
The plot of the Fourier integral fit is red.

Are the results from the integral fit and the fast fit the same? Are they significantly different?
How do you account for this?
What do you think would happen if you raised m = n + 1 to a much larger number?
How does your experience with these calculations mesh with what you said in part i)?

G.6) The transfer function is the Laplace transform of yunitimpluse[t]
The Laplace transform of yzeroinput[t] is the transfer function times the Laplace transform of the forcing function

Given an oscillator

$y''[t] + b\,y'[t] + c\,y[t] = f[t]$

with given starting data, lots of folks solve for y[t] by putting

$y[t] = \text{yzeroinput}[t] + \text{yunforced}[t].$

Here

yzeroinput[t]

solves

$y''[t] + b\,y'[t] + c\,y[t] = f[t]$

with $y[0] = 0$ and $y'[0] = 0$

and

yunforced[t]

solves

$y''[t] + b\,y'[t] + c\,y[t] = 0$

with the given starting data.

Coming up with the formula for yunforced[t] is always easy; it's just algebra which Mathematica can always handle.

Coming up with a precise formula for yzeroinput[t] is where all hell can break loose because of intractable integrals or intractable Laplace transforms. Engineers know how to make an end-run around this potential roadblock. They go with the oscillator diffeq:

```
Clear[y, t, f, b, c]
oscdiffeq = y''[t] + b y'[t] + c y[t] == f[t]
```

$c\,y[t] + b\,y'[t] + y''[t] == f[t]$

And then they write down the Laplace transform of yzeroinput:

```
ystarter = 0; yprimestarter = 0; Clear[Yzeroinput, s]
Yzeroinput[s_] = 1/(s^2 + b s + c) (LaplaceTransform[f[t], t, s] +
    b ystarter + s ystarter + yprimestarter)
```

$\frac{\text{LaplaceTransform}[f[t], t, s]}{c + b\,s + s^2}$

This tells you that the Laplace transform of yzeroinput[t] is

$\frac{1}{s^2 + b\,s + c}$ times the Laplace transform of f[t].

For this reason, some folks like to call

$H[s] = \frac{1}{s^2 + b\,s + c}$

by the name "transfer function."

```
Clear[H]
H[s_] = 1/(s^2 + b s + c)
```

$\frac{1}{c + b\,s + s^2}$

Again, the big point is that the Laplace transform of yzeroinput[t] is just:

```
H[s] LaplaceTransform[f[t], t, s]
```

$\frac{\text{LaplaceTransform}[f[t], t, s]}{c + b\,s + s^2}$

And you get this without an explicit calculation of yzeroinput[t].

□G.6.a)

How is the transfer function

$H[s] = \frac{1}{s^2 + b\,s + c}$

related to the characteristic equation for the unforced oscillator coming from

$y''[t] + b\,y'[t] + c\,y[t] = 0?$

How are singularities of H[s] related to roots of the characteristic equation?

What information do you get about an unforced oscillator when you know that all the singularities of its transfer function H[s] have negative real parts?

□G.6.b.i) The transfer function $\mathbf{H[s]} = \frac{1}{s^2 + b\,s + c}$ is the Laplace transform

of yunitimpulse[t]

Given the forced oscillator:

```
Clear[y, t, f, b, c]
oscdiffeq = y''[t] + b y'[t] + c y[t] == f[t]
```

$c\,y[t] + b\,y'[t] + y''[t] == f[t]$

You get yunitimpulse[t] as the solution of:

```
Clear[y, t, f, b, c]
unforcedoscdiffeq = y''[t] + b y'[t] + c y[t] == 0
```

$c\,y[t] + b\,y'[t] + y''[t] == 0$

with $y[0] = 0$ and $y'[0] = 1$.

The Laplace transform of yunitimpulse[t] is::

```
ystarter = 0; yprimestarter = 1;
Clear[Yunitimpulse, s]
Yunitimpulse[s_] = 1/(s^2 + b s + c)
  (LaplaceTransform[0, t, s] + b ystarter + s ystarter + yprimestarter)
```

$\frac{1}{c + b\,s + s^2}$

This is a good fringe benefit because it tells you that you can write down the Laplace transform of yunitimpulse[t] without going to all the trouble of calculating yunitimpulse[t].

How is the Laplace transform of yunitimpulse[t] related to the transfer function H[s] of the unforced oscillator diffeq

$y''[t] + b\,y'[t] + c\,y[t] = 0?$

What function has H[s] as its Laplace transform?

□G.6.b.ii)

Is it true that the Laplace transform of the solution of

```
Clear[y, t, f, b, c]
oscdiffeq = y''[t] + b y'[t] + c y[t] == f[t]
```

$c\,y[t] + b\,y'[t] + y''[t] == f[t]$

with $y[0] = 0$ and $y'[0] = 0$ is
the Laplace transform of yinitimpulse[t] times the Laplace transform of f[t]?

□G.6.b.iii)

Remember the convolution integral method. In this method you get yzeroinput[t] by calculating this convolution integral:

$$\text{yzeroinput}[t] = \int_0^t \text{yunitimpulse}[t-x]\, f[x]\, dx.$$

Let Mathematica calculate the Laplace transform of yzeroinput[t]:

```
Clear[yzeroinut, yunitimpulse, t, f, x]
yzeroinput[t_] = ∫_0^t yunitimpulse[t - x] f[x] dx;
LaplaceTransform[yzeroinput[t], t, s]
```

```
LaplaceTransform[f[t], t, s] LaplaceTransform[yunitimpulse[t], t, s]
```

How does this calculation jibe with what you said in part ii above?

G.7) *Mathematica* Fats and the Laplace transform

Some of the problems discussed here were taken from actual student assignments in an
electrical engineering course in Linear Signals and Systems at a well-known university.

Given a wide berth by the fraternity and sorority crowd because he constantly chews snuff right in the lab, "Mathematica Fats" is not at the social center of the Mathematica lab. Maybe this has something to do with his habit of popping Life Savers and drinking coffee without spitting out his Copenhagen snuff. You know the type. But recently Fats has become very popular with the electrical engineering crowd, and in fact has become one of the most popular residents of the Mathematica lab.
The reasons:
→ Fats knows mathematics;
→ Fats knows Mathematica; in fact he learned a lot of mathematics by exploring Mathematica, and
→ Fats knows how to use Mathematica to fake hand work.
This makes him especially useful to the students in Electrical Engineering 313 (Signals) whose professor demands that they do all of their work by hand.
Today you are sitting in on one of Fats' sessions in the lab.

□G.7.a) Trig identities

The first problem on the Electrical Engineering assignment is to find the Laplace transform of the function f[t] given by
$f[t] = \text{Sin}[t + a]$.
Fats' reaction:

```
Clear[F, s, t, a]
F[s_] = LaplaceTransform[Sin[t + a], t, s]
```

$$\frac{\text{Cos}[a] + s\,\text{Sin}[a]}{1 + s^2}$$

One student asks, "How do I make this look as if it were done by hand?"
Fats says, "Apply a trig identity like this:

```
handstep1 = Chop[TrigExpand[Sin[t + a]]]
```

```
Cos[t] Sin[a] + Cos[a] Sin[t]
```

And then replace Sin[t] and Cos[t] by their Laplace transforms:

```
LaplaceTransform[Sin[t], t, s]
```

$$\frac{1}{1 + s^2}$$

```
LaplaceTransform[Cos[t], t, s]
```

$$\frac{s}{1 + s^2}$$

Which you write up as if you looked them up in a table."

```
handstep1 /. {Sin[t] -> LaplaceTransform[Sin[t], t, s],
      Cos[t] -> LaplaceTransform[Cos[t], t, s]}
```

$$\frac{\text{Cos}[a]}{1 + s^2} + \frac{s\,\text{Sin}[a]}{1 + s^2}$$

Taking a fresh dip and placing it between his cheek and gum, Fats says, "You know this is really a problem from high school trig."
Now you try one.

Fake a hand calculation of the Laplace transform of Cos[t − 5.2].

□G.7.b) Using the definition of the Laplace transform

The second problem on the Electrical Engineering assignment is to find the Laplace transform of the function y[t] given by
$y[t] = \text{Sin}[t]$ for $0 \le t \le \pi$
but
$y[t] = 0$ for $\pi < t$.
Fats' reaction: Do this integral by Mathematica:

```
Clear[Y, y, s, t]
Y[s_] = ∫_0^π Sin[t] E^(-s t) dt
```

$$\frac{1}{1 + s^2} + \frac{E^{-\pi s}}{1 + s^2}$$

And then say you looked it up in a table."
Explain why Fats is right.

□G.7.c) Differentiating the Laplace transform

The third problem on the Electrical Engineering assignment is to find the Laplace transform of the function f[t] given by
$f[t] = t\,E^{-3t}\,\text{Sin}[4t]$
Fats' reaction:

```
Clear[F, s]
F[s_] = LaplaceTransform[t E^(-3 t) Sin[4 t], t, s]
```

$$\frac{8\,(3 + s)}{(16 + (3 + s)^2)^2}$$

"To make this look like it was done by hand, look at this:" says Fats.

```
Clear[g, t]
LaplaceTransform[t g[t], t, s]
```

$$-\text{LaplaceTransform}^{(0,0,1)}[g[t], t, s]$$

"This tells you that the Laplace transform of t g[t] is minus the derivative of the Laplace transform of g[t].
Now say you looked up this in a table:

```
LaplaceTransform[E^(-3 t) Sin[4 t], t, s]
```

$$\frac{4}{16 + (3 + s)^2}$$

Now differentiate with respect to s and multiply by −1 to get the Laplace transform of

$f[t] = t\,E^{-3t}\,\text{Sin}[4t]$."
Do it and check yourself with Mathematica.

□G.7.d) The Laplace transform of the convolution is the product of the Laplace transforms.

The next problem on the EE 313 homework set is to find the inverse Laplace transform of:

```
Clear[Y, s]
Y[s_] = 1/((s^2 + 1) (s^2 + 4))
```

$$\frac{1}{(1 + s^2)\,(4 + s^2)}$$

Some of the engineering students are clueless. Fats says, "That prof probably wanted you to do some miserable algebra. Let's surprise the prof. Look at this."

```
Clear[f, g, h, t, s, x, y]
h[t_] = ∫_0^t f[t - x] g[x] dx
```

$$\int_0^t f[t-x]\, g[x]\, dx$$

```
LaplaceTransform[h[t], t, s]
```

```
LaplaceTransform[f[t], t, s] LaplaceTransform[g[t], t, s]
```

Fats goes on: "This tells you that the Laplace transform of the convolution

$$h[t] = \int_0^t f[t-x]\, g[x]\, dx$$

is the Laplace transform of f[t] times the Laplace transform of g[t].
That's cool.
So you can get the inverse transform of
$Y[s] = \frac{1}{(s^2+1)(s^2+4)}$
by looking up the inverse transforms of
$\frac{1}{s^2+1}$ and $\frac{1}{s^2+4}$
in a table like this :

```
Clear[f, t];
f[t_] = InverseLaplaceTransform[1/(s^2 + 1), s, t]
```

```
Sin[t]
```

```
Clear[g, t]
g[t_] = InverseLaplaceTransform[1/(s^2 + 4), s, t]
```

$\frac{1}{2}\,\mathrm{Sin}[2\,t]$

And now the function y[t] whose Laplace transform is Y[s] is just:

```
Clear[y, t]
y[t_] = ∫_0^t f[t - x] g[x] dx
```

$\frac{\mathrm{Sin}[t]}{3} - \frac{1}{6}\,\mathrm{Sin}[2\,t]$

Check:

```
Together[LaplaceTransform[y[t], t, s]]
```

$\frac{1}{(1+s^2)\,(4+s^2)}$

This is the same as:

```
Y[s]
```

$\frac{1}{(1+s^2)\,(4+s^2)}$

Now you try one.

Use the same method to come up with the inverse Laplace Transform of

$$Y[s] = \frac{s}{(s+2)\,(s^2+3)} = \frac{1}{s+2}\;\frac{s}{(s^2+3)}$$

□G.7.e) Mathematica Fats and Calculus Cal

There is some pressure to have Calculus Cal join Fats' group. But try as they might, the students could not convince Calculus Cal to get anywhere near Mathematica Fats. Cal knows that when he blurts out something really stupid such as
"If F[s] is the Laplace transform of f[t] and G[s] is the Laplace transform of g[t], then
F[s] G[s]
is the Laplace transform of
f[t] g[t],"
then Fats will eat him up and spit him out with his spent snuff.

Just to keep yourself at a big distance from Cal, say why Cal's idea that if F[s] is the Laplace transform of f[t] and G[s] is the Laplace transform of g[t], then F[s] G[s] is the Laplace transform of f[t] g[t] is really stupid.
What function has F[s] G[s] for its Laplace transform?

□G.7.f) Partial fractions

The next problem on the EE 313 homework set is to find the inverse Laplace transform of:

```
Clear[Y, s, t]
Y[s_] = (6 s^2 + 22 s + 18)/((s + 1) (s + 2) (s + 3))
```

$\frac{18 + 22\,s + 6\,s^2}{(1+s)\,(2+s)\,(3+s)}$

Fats' first reaction is:

```
InverseLaplaceTransform[Y[s], s, t]
```

$3\,E^{-3\,t} + 2\,E^{-2\,t} + E^{-t}$

Fats says, "Here's how to make it look like a hand calculation:"

```
Apart[Y[s]]
```

$\frac{1}{1+s} + \frac{2}{2+s} + \frac{3}{3+s}$

Fancy folks call this the partial fraction decomposition of Y[s].
Partial fractions are a big deal in old-fashioned calculus courses.

A student says, "What's that?"
Fats says, "It's the same as Y[s]:"

```
Simplify[Apart[Y[s]] - Y[s]]
```

0

Then Fats says to look at this:

```
LaplaceTransform[E^(k t), t, s]
```

$\frac{1}{-k+s}$

Now you take over. Explain where the inverse Laplace transform of Y[s] comes from.

□G.7.g) Periodic impulses

Another problem is to find the inverse Laplace transform of

$$Y[s] = \frac{E^s}{E^s - 1}$$

```
InverseLaplaceTransform[E^s/(E^s - 1), s, t]
```

$\mathrm{InverseLaplaceTransform}\left[\frac{E^s}{-1+E^s}, s, t\right]$

No dice.
The students run to Fats who says: "Not surprising; look at this:"

```
Clear[x]
Series[1/(1 - x), {x, 0, 9}]
```

$1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + O[x]^{10}$

So

$$\frac{E^s}{E^s-1} = \frac{1}{1-E^{-s}}$$
$$= 1 + E^{-s} + E^{-2\,s} + E^{-3\,s} + \ldots E^{-k\,s} + \ldots$$

Fats continues: "Now look at this:"

```
Clear[s, t, k]
LaplaceTransform[DiracDelta[t], t, s]
```

1

```
LaplaceTransform[DiracDelta[t - 1], t, s]
```

E^{-s}

```
LaplaceTransform[DiracDelta[t - 2], t, s]
```

$E^{-2\,s}$

```
LaplaceTransform[DiracDelta[t - 3], t, s]
```

$E^{-3\,s}$

```
LaplaceTransform[DiracDelta[t - 10], t, s]
```

$E^{-10\,s}$

Taking a new dip of snuff and a huge slug of coffee, Fats licks his lips and says , "It should now be clear what periodic impulse has

$$\frac{E^s}{E^s-1} = \frac{1}{1-E^{-s}}$$
$$= 1 + E^{-s} + E^{-2\,s} + E^{-3\,s} + \ldots E^{-k\,s} + \ldots$$

for its Laplace transform."

What does Fats mean?

Fats goes on to say, "I think a rather slick answer to this problem is
DiracDelta[Sin[π t]]."
Is Fats right?

The characters Mathematica Fats and Calculus Cal are based on real people.

G.8) Critically damped oscillators

□G.8.a)

Here's a damped oscillator whose characteristic equation has only one solution:

```
Clear[r];
Clear[y, yoriginal, t]
originaloscillator = y''[t] + (2 r) y'[t] + r^2 y[t] == 0
```

$r^2\,y[t] + 2\,r\,y'[t] + y''[t] == 0$

```
Clear[z]
charequation =
 originaloscillator /. {y''[t] -> z^2, y'[t] -> z, y[t] -> 1};
zsols = Solve[charequation, z]
```

{{z → -r}, {z → -r}}

Here's a formula for the exact solution of this oscillator with starting data
y[0] = p and y′[0] = q.

```
Clear[p, q];
originalsol =
DSolve[{originaloscillator,
    y[0] == p, y'[0] == q}, y[t], t];

Clear[yoriginal];
yoriginal[t_] = y[t] /. originalsol[[1]]
```

$$E^{-r\,t}\,(p + (q + p\,r)\,t)$$

Why are you not surprised by the extra t on the right?

□G.8.b.i)

Here is a damped oscillator whose characteristic equation has only one solution:

```
Clear[ycritical];
r = 3/4;
criticaloscillator = y''[t] + (2 r) y'[t] + r^2 y[t] == 0
```

$$\frac{9\,y[t]}{16} + \frac{3\,y'[t]}{2} + y''[t] == 0$$

```
Clear[p, q]
criticalsol =
 DSolve[{criticaloscillator, y[0] == p, y'[0] == q}, y[t], t];
ycritical[t_] = y[t] /. criticalsol[[1]]
```

$$E^{-3\,t/4}\left(p + \left(\frac{3\,p}{4} + q\right)t\right)$$

Change the damping term a wee bit and keep everything else the same and you get:

```
Clear[ychangedamp]
changedamposcillator = y''[t] + (2 r + h) y'[t] + r^2 y[t] == 0
```

$$\frac{9\,y[t]}{16} + \left(\frac{3}{2} + h\right)y'[t] + y''[t] == 0$$

```
changedampsol =
DSolve[{changedamposcillator,
    y[0] == p, y'[0] == q}, y[t], t];

ychangedamp[t_, h_] = y[t] /. changedampsol[[1]]
```

$$-\frac{E^{\frac{1}{4}(-3-2h-2\sqrt{h}\sqrt{3+h})\,t}\left(\left(3+2h-2\sqrt{h}\sqrt{3+h}\right)p+4q\right)}{4\sqrt{h}\sqrt{3+h}} + \frac{E^{\frac{1}{4}(-3-2h+2\sqrt{h}\sqrt{3+h})\,t}\left(\left(3+2h+2\sqrt{h}\sqrt{3+h}\right)p+4q\right)}{4\sqrt{h}\sqrt{3+h}}$$

When you make h = 0.0001, you get:

```
ychangedamp[t, 0.0001]
```

$$-14.4335\,E^{-0.75871\,t}\,(2.96556\,p + 4\,q) + 14.4335\,E^{-0.74139\,t}\,(3.03484\,p + 4\,q)$$

When you make h = −0.0001, you get:

```
Chop[ComplexExpand[ychangedamp[t, -0.0001]]]
```

$$1.\,E^{-0.74995\,t}\,p\,\mathrm{Cos}[0.00866011\,t] + 86.5982\,E^{-0.74995\,t}\,p\,\mathrm{Sin}[0.00866011\,t] + 115.472\,E^{-0.74995\,t}\,q\,\mathrm{Sin}[0.00866011\,t]$$

Compare all three formulas:

```
ychangedamp[t, 0.0001]
```

$$-14.4335\,E^{-0.75871\,t}\,(2.96556\,p + 4\,q) + 14.4335\,E^{-0.74139\,t}\,(3.03484\,p + 4\,q)$$

```
ycritical[t]
```

$$E^{-3\,t/4}\left(p + \left(\frac{3\,p}{4} + q\right)t\right)$$

```
Chop[ComplexExpand[ychangedamp[t, -0.0001]]]
```

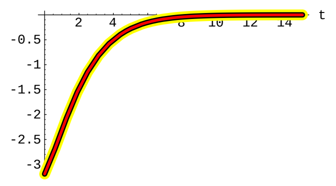
$$1.\,E^{-0.74995\,t}\,p\,\mathrm{Cos}[0.00866011\,t] + 86.5982\,E^{-0.74995\,t}\,p\,\mathrm{Sin}[0.00866011\,t] + 115.472\,E^{-0.74995\,t}\,q\,\mathrm{Sin}[0.00866011\,t]$$

Account for the change in character of the formulas.

□G.8.b.ii)

Now go with random p and q and see how the three formulas plot out:

```
p = Random[Real, {-5, 5}];
q = Random[Real, {-5, 5}]; Plot[{ychangedamp[t, 0.0001],
   ycritical[t], Chop[ComplexExpand[ychangedamp[t, -0.0001]]]},
  {t, 0, 15}, PlotStyle -> {{Thickness[0.04], Yellow},
    {Thickness[0.02], Black}, {Thickness[0.01], Red}},
  AxesLabel -> {"t", ""}, PlotRange -> All, AspectRatio -> 1/GoldenRatio];
```



Rerun several times.

Explain this:
Even though the formulas for all three were radically different in character, all three plot out approximately the same.

□G.8.b.iii)

Here are the three formulas that you plotted out above:

```
ychangedamp[t, 0.0001]
```

$$97.2936\,E^{-0.75871\,t} - 100.486\,E^{-0.74139\,t}$$

```
ycritical[t]
```

$$E^{-3\,t/4}\,(-3.19219 - 1.71269\,t)$$

```
Chop[ComplexExpand[ychangedamp[t, -0.0001]]]
```

$$-3.19219\,E^{-0.74995\,t}\,\mathrm{Cos}[0.00866011\,t] - 197.749\,E^{-0.74995\,t}\,\mathrm{Sin}[0.00866011\,t]$$

One of these actually oscillates (just a wee bit) and the others do not.
Which one actually oscillates (in theory)?

□G.8.b.iv)

Lots of folks say that a damped oscillator whose characteristic equation has only one solution is "critically damped" in the sense that
→ If the damping term is decreased, then the resulting oscillator actually oscillates.
→ If the damping term is increased, then the resulting oscillator is "overdamped" because it does not oscillate at all.
Why do you think folks say this?

G.9) Recovering the expansion of a function from its Laplace transform

□G.9.a.i)

By now, you know that you can bypass a lot of grubby calculation provided you deal exclusively in Laplace transforms. This may seem strange, but it's not at all dangerous because the Laplace transform of a function carries the full DNA of the function.
For instance, if you have the Laplace transform Y[s] of a function y[t], you can recover the values of y[0] and y′[0] directly from Y[s] without inverting. For instance, to recover y[0], you just calculate

y[0] = Limit[s Y[s], s → ∞]

And, once you have y[0] you just calculate

y′[0] = Limit[s^2 Y[s] − s y[0], s → ∞]

Try these out on this sample Y[s]:

```
Clear[y, s, t, Y];
Y[s_] = 2 (4 + s)/(9 + (s - 5)^2)
```

$$\frac{2\,(4+s)}{9+(-5+s)^2}$$

To recover y[0] without inverting Y[s], look at

y[0] = Limit[s Y[s], s → ∞]

```
ExpandAll[Together[s Y[s]]]
```

$$\frac{8\,s}{34-10\,s+s^2} + \frac{2\,s^2}{34-10\,s+s^2}$$

By going with quotients of the dominant terms, you can pick off

y[0] = Limit[s Y[s], s → ∞]

$$= \mathrm{Limit}\left[\frac{8}{s} + \frac{2\,s^2}{s^2},\ s \to \infty\right]$$

$$= 0 + 2 = 2$$

To recover $y'[0]$ without inverting $Y[s]$, look at

$y'[0] = \text{Limit}[s^2 Y[s] - s\, y[0], s \to \infty]$:

```
Together[s^2 Y[s] - s y[0]] /. y[0] → 2
```

$$\frac{-68 s + 28 s^2}{34 - 10 s + s^2}$$

By going with quotients of the dominant terms, you can pick off

$$y'[0] = \text{Limit}[s^2 Y[s] - s\, y[0], s \to \infty]$$

$$= \text{Limit}\left[\frac{-68 s + 28 s^2}{34 - 10 s + s^2}, s \to \infty\right]$$

$$= \text{Limit}\left[\frac{28 s^2}{s^2}, s \to \infty\right] = 28.$$

Check:

```
Clear[y, t];
y[t_] = InverseLaplaceTransform[Y[s], s, t];
y[0] == 2
y'[0] == 28
```

```
True
True
```

This checks.
Now you do one.
Here is the Laplace transform of a function y[t]:

```
Clear[Y, s];
Y[s_] = (3 s^5 + 12 s^3 - 15 s)/(1 + 3 s^2 + 3 s^4 + s^6)
```

$$\frac{-15 s + 12 s^3 + 3 s^5}{1 + 3 s^2 + 3 s^4 + s^6}$$

Calculate the limits

$y[0] = \text{Limit}[s\, Y[s], s \to \infty]$

and

$y'[0] = \text{Limit}[s^2 Y[s] - s\, y[0], s \to \infty]$

to come up with y[0] and y′[0] without inverting the Laplace transform. Then invert the Laplace transform to check your calculations.

□G.9.a..ii) Explanations

Look at this calculation which was explained in the Basics:

```
Clear[y, s, t];
LaplaceTransform[y'[t], t, s]
```

```
s LaplaceTransform[y[t], t, s] - y[0]
```

Putting

$Y[s] = \text{LaplaceTransform}[y[t], t, s]$,

this calculation tells you that

$$s\, Y[s] - y[0] = \text{LaplaceTransform}[y'[t], t, s] = \int_0^\infty E^{-s t}\, y'[t]\, dt$$

So

$$y[0] = s\, Y[s] - \int_0^\infty E^{-s t}\, y'[t]\, dt$$

no matter what s is.
Take the limit as $s \to \infty$ to get

$$y[0] = \text{Limit}\left[s\, Y[s] - \int_0^\infty E^{-s t}\, y'[t]\, dt, s \to \infty\right]$$

$$= \text{Limit}[s\, Y[s], s \to \infty] - \text{Limit}\int_0^\infty E^{-s t}\, y'[t]\, dt, s \to \infty]$$

$$= \text{Limit}[s\, Y[s], s \to \infty] - 0$$

(because $E^{-s t}$ collapses as s gets large and positive)

$$= \text{Limit}[s\, Y[s], s \to \infty]$$

This explains why you can calculate

$y[0] = \text{Limit}[s\, Y[s], s \to \infty]$

Now look at this calculation which was also explained in the Basics:

```
Clear[y, s, t]
LaplaceTransform[y''[t], t, s]
```

```
s^2 LaplaceTransform[y[t], t, s] - s y[0] - y′[0]
```

This tells you that

$$s^2 Y[s] - s\, y[0] - y'[0] = \int_0^\infty E^{-s t}\, y''[t]\, dt.$$

Adapt the explanation above to explain why, once you have your hands on y[0], you can calculate

$y'[0] = \text{Limit}[s^2 Y[s] - s\, y[0], s \to \infty]$:

□G.9.a.iii)

Look at this:

```
Clear[y, s, t];
LaplaceTransform[y''[t], t, s]
```

```
s^2 LaplaceTransform[y[t], t, s] - s y[0] - y′[0]
```

Once you have your hands on y[0] and y′[0], what limit involving
Y[s], y[0] and y′[0]
do you calculate to calculate y″[0]?
No explanation is asked for.

□G.9.a.iv)

Look at this:

```
Clear[y, s, t]
LaplaceTransform[y'''[t], t, s]
```

```
s^3 LaplaceTransform[y[t], t, s] - s^2 y[0] - s y′[0] - y″[0]
```

Once you have your hands on y[0] , y′[0], and y″[0],what limit involving
Y[s], y[0] , y′[0]and y″[0]
do you calculate to calculate the third derivative y^{TagBox[(3)]}[0] ?
No explanation is asked for.

```
This part is for only those participants
who have a knowledge of series expansions.
```

At this point, it becomes clear that, given Y[s], you could go on and on to slam out as many of the values of the higher derivatives at 0
y[0], y′[0], y″[0], y$^{\text{TagBox[(3)]}}$[0], y$^{\text{TagBox[(4)]}}$[0],y$^{\text{TagBox[(5)]}}$[0], . . .y$^{\text{TagBox[(k)]}}$[0], ...
as you like. After you have these, you have as much of the expansion of y[t] in powers of t as you like:

```
Clear[y, t];
Series[y[t], {t, 0, 6}]
```

$$y[0] + y'[0]\, t + \frac{1}{2} y''[0]\, t^2 + \frac{1}{6} y^{(3)}[0]\, t^3 + \frac{1}{24} y^{(4)}[0]\, t^4 + \frac{1}{120} y^{(5)}[0]\, t^5 + \frac{1}{720} y^{(6)}[0]\, t^6 + O[t]^7$$

```
Clear[y, t];
Series[y[t], {t, 0, 12}]
```

$$y[0] + y'[0]\, t + \frac{1}{2} y''[0]\, t^2 + \frac{1}{6} y^{(3)}[0]\, t^3 + \frac{1}{24} y^{(4)}[0]\, t^4 + \frac{1}{120} y^{(5)}[0]\, t^5 + \frac{1}{720} y^{(6)}[0]\, t^6 + \frac{y^{(7)}[0]\, t^7}{5040} + \frac{y^{(8)}[0]\, t^8}{40320} + \frac{y^{(9)}[0]\, t^9}{362880} + \frac{y^{(10)}[0]\, t^{10}}{3628800} + \frac{y^{(11)}[0]\, t^{11}}{39916800} + \frac{y^{(12)}[0]\, t^{12}}{479001600} + O[t]^{13}$$

This tells you that if you assume that y[t] has an expansion in powers of t and you assume that all the higher derivatives of y[t] are legal for the Laplace transform, then you can recover the expansion of y[t] in powers of t. And this gives you a hint of why Laplace transforms can be inverted. And it gives you a hint of why you can leave the Laplace transform world anytime you like.

G.10) Using Fourier fit to slam out trig identities

Activate this code.

```
Clear[FastFourierfit, Fourierfitters, F, Fvalues, n, k,
   jump, num, numtab, coeffs, t, L];
jump[n_] := jump[n] = N[1/(2 n)];
Fvalues[F_, L_, n_] :=
 N[Table[F[L t], {t, 0, 1 - jump[n], jump[n]}]];


numtab[n_] := numtab[n] = Table[k, {k, 1, n}];


Fourierfitters[L_, n_, t_] := Table[E^(2 π I k t/L),
    {k, -n + 1, n - 1}];
coeffs[n_, list_] := Join[Reverse[Part[Fourier[list], numtab[n]]],
  Part[InverseFourier[list], Drop[numtab[n], 1]]] /
  N[Sqrt[Length[list]]]


FastFourierfit[F_, L_, n_, t_] :=
  Chop[Fourierfitters[L, n, t] . coeffs[n, Fvalues[F, L, n]]];
```

□G.10.a.i)

Look at these little experiments:

□Experiment 1: f[t] = Sin[t]^2

```
Clear[f, t]
f[t_] = Sin[t]^2;
L = 2 π;
 {n, Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]}
 {n, 1, 8}
{1, 0}
{2, 0.5}
{3, 0.5 - 0.5 Cos[2 t]}
{4, 0.5 - 0.5 Cos[2 t]}
{5, 0.5 - 0.5 Cos[2 t]}
{6, 0.5 - 0.5 Cos[2 t]}
{7, 0.5 - 0.5 Cos[2 t]}
{8, 0.5 - 0.5 Cos[2 t]}
```

Compare:

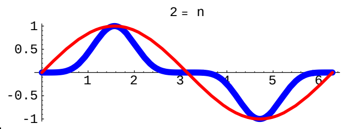
```
N[Expand[TrigReduce[f[t]]]]
0.5 - 0.5 Cos[2. t]
```

□Experiment 2: f[t] = Cos[t]^4:

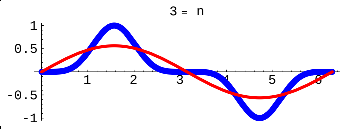
```
Clear[f, t]
f[t_] = Cos[t]^4;
L = 2 π;
 {n, Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]}
 {n, 1, 8}
{1, 1.}
{2, 0.5}
{3, 0.375 + 0.625 Cos[2 t]}
{4, 0.375 + 0.5 Cos[2 t]}
{5, 0.375 + 0.5 Cos[2 t] + 0.125 Cos[4 t]}
{6, 0.375 + 0.5 Cos[2 t] + 0.125 Cos[4 t]}
{7, 0.375 + 0.5 Cos[2 t] + 0.125 Cos[4 t]}
{8, 0.375 + 0.5 Cos[2 t] + 0.125 Cos[4 t]}
```

Compare:

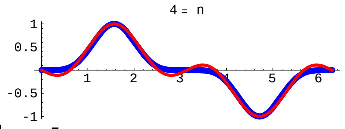
```
N[Expand[TrigReduce[f[t]]]]
0.375 + 0.5 Cos[2. t] + 0.125 Cos[4. t]
```



```
n = 3;
Clear[fitter];
fitter[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];

Plot[{f[t], fitter[t]}, {t, 0, L},
  PlotStyle -> {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  PlotRange -> All, PlotLabel -> n "= n"];
```



```
n = 4;
Clear[fitter];
fitter[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];

Plot[{f[t], fitter[t]}, {t, 0, L},
  PlotStyle -> {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  PlotRange -> All, PlotLabel -> n "= n"];
```



```
n = 5;
Clear[fitter];
fitter[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];

Plot[{f[t], fitter[t]}, {t, 0, L},
  PlotStyle -> {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  PlotRange -> All, PlotLabel -> n "= n"];
```

□Experiment 3:f[t] = Sin[t]^8

```
Clear[f, t]
f[t_] = Sin[t]^8;
L = 2 π;
 {n, Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]}
 {n, 1, 12}
{1, 0}
{2, 0.5}
{3, 0.210938 - 0.210938 Cos[2 t]}
{4, 0.28125 - 0.5 Cos[2 t]}
{5, 0.273437 - 0.429687 Cos[2 t] + 0.15625 Cos[4 t]}
{6, 0.273438 - 0.4375 Cos[2 t] + 0.226562 Cos[4 t]}
{7, 0.273438 - 0.4375 Cos[2 t] + 0.21875 Cos[4 t] - 0.0546875 Cos[6 t]}
{8, 0.273437 - 0.4375 Cos[2 t] + 0.21875 Cos[4 t] - 0.0625 Cos[6 t]}
{9, 0.273438 - 0.4375 Cos[2 t] + 0.21875 Cos[4 t] - 0.0625 Cos[6 t] + 0.0078125
{10, 0.273437 - 0.4375 Cos[2 t] + 0.21875 Cos[4 t] - 0.0625 Cos[6 t] + 0.007812
{11, 0.273437 - 0.4375 Cos[2 t] + 0.21875 Cos[4 t] - 0.0625 Cos[6 t] + 0.007812
{12, 0.273438 - 0.4375 Cos[2 t] + 0.21875 Cos[4 t] - 0.0625 Cos[6 t] + 0.007812
```

Compare:

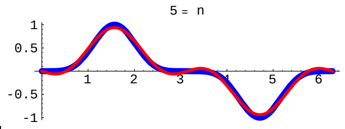
```
N[Expand[TrigReduce[f[t]]]]
0.273438 - 0.4375 Cos[2. t] + 0.21875 Cos[4. t] - 0.0625 Cos[6. t] +
 0.0078125 Cos[8. t]
```

Do you have any hunch about what's going on here, or are you clueless?
If you have a hunch, then say what your hunch is.
If you are sure you know what's going on, then explain in full.

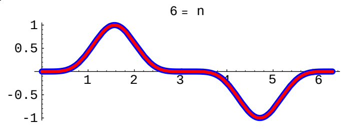
Click for a tip.

To get a good clue, you might want to look at plots like these:

```
Clear[f, n, t];
f[t_] = Sin[t]^5;
L = 2 Pi;

n = 2;
Clear[fitter];
fitter[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];

Plot[{f[t], fitter[t]}, {t, 0, L},
  PlotStyle -> {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  PlotRange -> All, PlotLabel -> n "= n"];
```



```
n = 6;
Clear[fitter];
fitter[t_] = Chop[ComplexExpand[FastFourierfit[f, L, n, t]]];

Plot[{f[t], fitter[t]}, {t, 0, L},
  PlotStyle -> {{Thickness[0.02], Blue}, {Thickness[0.01], Red}},
  PlotRange -> All, PlotLabel -> n "= n"];
```



□G.10.a.ii)

Run this new experiment:

```
Clear[f, t];
f[t_] = Cos[t]^20;
L = 2 π;
 {n, Chop[ComplexExpand[FastFourierfit[f, L, n, t]]]}
 {n, 1, 12}
{1, 1.}
{2, 0.5}
{3, 0.333334 + 0.666666 Cos[2 t]}
{4, 0.250488 + 0.5 Cos[2 t]}
{5, 0.20577 + 0.403566 Cos[2 t] + 0.390663 Cos[4 t]}
{6, 0.185438 + 0.352104 Cos[2 t] + 0.314562 Cos[4 t]}
{7, 0.178371 + 0.329962 Cos[2 t] + 0.269878 Cos[4 t] + 0.221788 Cos[6 t]}
{8, 0.176559 + 0.322571 Cos[2 t] + 0.249512 Cos[4 t] + 0.177429 Cos[6 t]}
{9, 0.176235 + 0.320723 Cos[2 t] + 0.242443 Cos[4 t] + 0.157099 Cos[6 t] + 0.10
{10, 0.176199 + 0.320396 Cos[2 t] + 0.240631 Cos[4 t] + 0.150032 Cos[6 t] + 0.0
{11, 0.176197 + 0.32036 Cos[2 t] + 0.240307 Cos[4 t] + 0.14822 Cos[6 t] + 0.076
{12, 0.176197 + 0.320358 Cos[2 t] + 0.240271 Cos[4 t] + 0.147896 Cos[6 t] + 0.0
```

Compare:

```
N[Expand[TrigReduce[f[t]]]]
```

```
0.176197 + 0.320358 Cos[2. t] + 0.240269 Cos[4. t] +
 0.147858 Cos[6. t] + 0.0739288 Cos[8. t] + 0.0295715 Cos[10. t] +
 0.0092411 Cos[12. t] + 0.00217438 Cos[14. t] + 0.000362396 Cos[16. t] +
 0.000038147 Cos[18. t] + 1.90735 × 10^-6 Cos[20. t]
```

What's your opinion on the reason that this experiment didn't turn out the way the experiments in part i) above turned out?
Modify this experiment so that it does turn out the way the experiments in part i) above turned out.

□G.10.a.iii)

Go with f[t] = Sin[t]$^{12.}$
And look at:

```
Clear[f, t];
f[t_] = Sin[t]^12;
N[Expand[TrigReduce[f[t]]]]
```

```
0.225586 - 0.386719 Cos[2. t] + 0.241699 Cos[4. t] - 0.107422 Cos[6. t] +
 0.0322266 Cos[8. t] - 0.00585938 Cos[10. t] + 0.000488281 Cos[12. t]
```

Use Fourier fit to slam out the same trig identity.

□G.10.a.iv)

Go with f[t] = Sin[5 t] Sin[3 t].
And look at:

```
Clear[f, t];
f[t_] = Sin[5 t] Sin[3 t];
N[Expand[TrigReduce[f[t]]]]
```

```
0.5 Cos[2. t] - 0.5 Cos[8. t]
```

Use fast Fourier fit to slam out the same trig identity.

□G.10.a.v)

Go with f[t] = Cosh[I t]
And look at:

```
Clear[f, t];
f[t_] = Cosh[I t];
N[Expand[TrigReduce[f[t]]]]
```

```
Cos[t]
```

Use Fourier fit to slam out the same trig identity.

□G.10.a.vi)

Go with f[t] = Sinh[I t]
And look at:

```
Clear[f, t];
f[t_] = Sinh[I t];
N[Expand[f[t], Trig → True]]
```

```
1. I Sin[t]
```

Use Fourier fit to slam out the same trig identity.