Math 2280-2

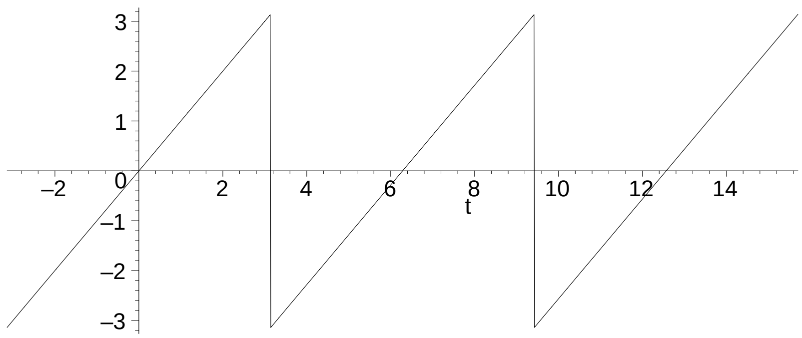
Friday April 6, 2001

Forced oscillations: resonance mysteries

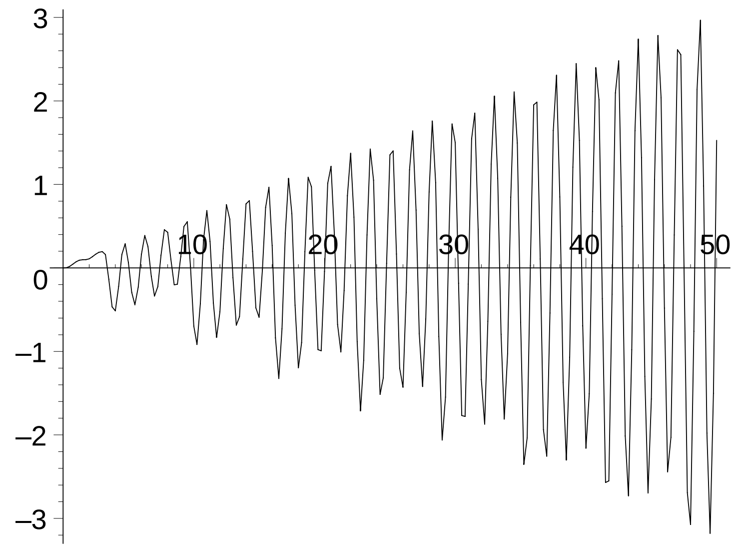
```
> restart:
  with(plots):
  with(DEtools):
Warning, the name changecoords has been redefined
>
> f1:=x->evalf(2*Pi*(frac((x+Pi)/(2*Pi)))-Pi);  #sawtooth!
```

$$f1 := x \to \mathrm{evalf}\left(2\,\pi\,\mathrm{frac}\left(\frac{1}{2}\,\frac{x+\pi}{\pi}\right)-\pi\right)$$

```
> plot(f1(t),t=-Pi..5*Pi,color=black);
  #has minimum period = 2*Pi
```




```
> spring:=dsolve({diff(x(t),t)=y(t),diff(y(t),t)=-16*x(t)+f1(t),
  x(0)=0,y(0)=0},{x(t),y(t)},type=numeric):
  #has natural angular frequency of 4, so natural period of Pi/2
> odeplot(spring,[t,x(t)],0..50,numpoints=200,color=black);
  #forced oscillations
```

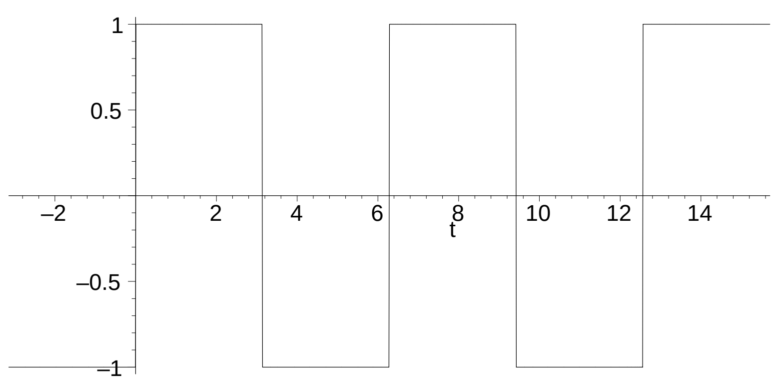


That sure looks like resonance, even though the natural angular frequency of our system ‘‘4’’ and the angular frequency of the forcing function is ‘‘1’’.   MYSTERY!!!!!!!

```
> f2:=t->f1(t)/(abs(f1(t)+.00000001));  #square wave
  #with smallest period of 2*Pi
```

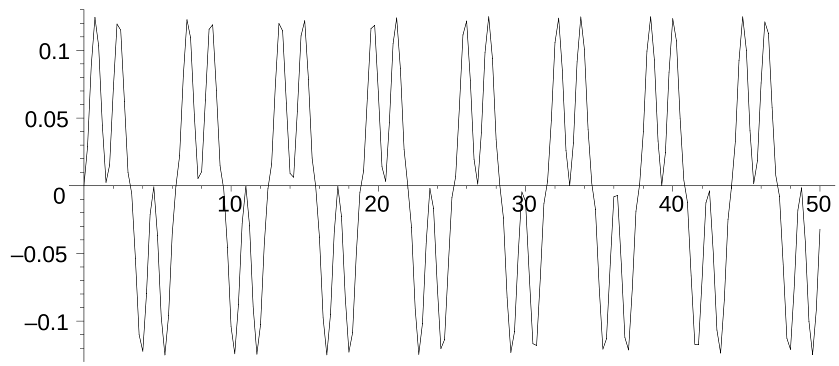
$$f2 := t \rightarrow \frac{\mathrm{f1}(t)}{\left|\mathrm{f1}(t) + .1\ 10^{-7}\right|}$$

```
> plot(f2(t),t=-Pi..5*Pi,color=black);
```




```
> spring2:=dsolve({diff(x(t),t)=y,diff(y(t),t)=-16*x(t)+f2(t),x(0)=0
  ,y(0)=0},{x(t),y(t)},type=numeric):
```

```
> odeplot(spring2,[t,x(t)],0..50,numpoints=200, color=black);
```




```
>
```

Now there's no resonance!!!! what gives?

Part 2: Fourier approximations

```
> restart:with(linalg):with(plots):
      #we'll use linear algebra and plots below
Warning, the protected names norm and trace have been redefined and unprotected

Warning, the name changecoords has been redefined
```

On the interval [-Pi..Pi] the functions {1,cos(x),cos(2x),...sin(x),sin(2x),sin(3x),...} are mutually orthogonal with respect to the inner product

```
> Dot:=(f,g)->int(f(x)*g(x),x=-Pi..Pi);
```

$$Dot := (f, g) \to \int_{-\pi}^{\pi} f(x)\, g(x)\, dx$$

```
> Mag:=f->sqrt(Dot(f,f)); #computes the magnitude of
       #a vector
  Dist:=(f,g)->Mag(f-g);  #computes the ''distance'' between
       #two vectors.
  angle:=(f,g)->arccos(Dot(f,g)/(Mag(f)*Mag(g)));
       #computes the ''angle'' between two vectors
```

$$Mag := f \to \sqrt{\mathrm{Dot}(f, f)}$$

$$Dist := (f, g) \to \mathrm{Mag}(f - g)$$

$$angle := (f, g) \to \arccos\left(\frac{\mathrm{Dot}(f, g)}{\mathrm{Mag}(f)\,\mathrm{Mag}(g)}\right)$$

Verify that sin(3*x) and sin(4*x) are orthogonal, and compute their magnitudes:

```
> Dot(x->sin(3*x),x->sin(4*x));
  Mag(x->sin(3*x));Mag(x->sin(4*x));
```

$$0$$

$$\sqrt{\pi}$$

$$\sqrt{\pi}$$

When we do the projection of a function f onto Vn, the coefficients for each basis vector are just the inner product of that basis vector with f, divided by the square of the magnitude of the vector Let's illustrate by projecting the absolute value function onto W10:

```
> f:=x->abs(x);  #we'll do Fourier for the absolute value function
  n:=10;   #order 10 expansion
  a:=vector(n); #cos coefficients
  b:=vector(n);  #sin coefficients
  a0:=(1/Pi)*Dot(f,1);  #order zero Fourier coefficient
>
```

$$f := \text{abs}$$

$$n := 10$$

$$a := \text{array}(1 .. 10, [\,])$$

$$b := \text{array}(1 .. 10, [\,])$$

$$a0 := \pi$$

```
> for i from 1 to n do  #compute the projection coefficients
  b[i]:=(1/Pi)*Dot(f,x->sin(i*x));
  a[i]:=(1/Pi)*Dot(f,x->cos(i*x));
  od:
```

```
> evalm(b);#why will an even function have Fourier sine
    #coefficients all equal to zero?
```

$$[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

The answer to the rhetorical question above is that the product of an even function with an odd function is an odd function. We already asserted that the integral of an odd function over an interval [-L,L] is always zero. By the way, could you prove that assertion?

```
> evalm(a); #Could you predict higher order coefficients
      #from the pattern you see here?
```

$$\left[-4\frac{1}{\pi}, 0, -\frac{4}{9}\frac{1}{\pi}, 0, -\frac{4}{25}\frac{1}{\pi}, 0, -\frac{4}{49}\frac{1}{\pi}, 0, -\frac{4}{81}\frac{1}{\pi}, 0\right]$$

The answer to this rhetorical question is yes: the i=even cos coefficients are zero, the i=odd ones are -4 divided by the product of Pi with i^2.

```
> approx:=x->sum(b[k]*sin(k*x),k=1..n)
             +sum(a[m]*cos(m*x),m=1..n)
             +a0/2;
```

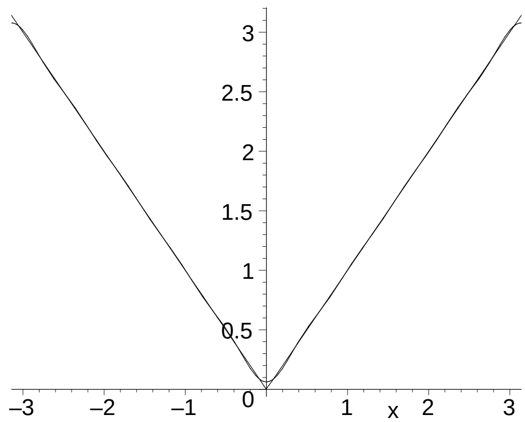
$$approx := x \to \left(\sum_{k=1}^{n} b_k \sin(k\,x)\right) + \left(\sum_{m=1}^{n} a_m \cos(m\,x)\right) + \frac{1}{2}\,a0$$

```
> approx(x); #order 10 Fourier expansion for abs(x):
```

$$-4\frac{\cos(x)}{\pi} - \frac{4}{9}\frac{\cos(3\,x)}{\pi} - \frac{4}{25}\frac{\cos(5\,x)}{\pi} - \frac{4}{49}\frac{\cos(7\,x)}{\pi} - \frac{4}{81}\frac{\cos(9\,x)}{\pi} + \frac{1}{2}\pi$$

```
> almost:=plot(approx(x),x=-Pi..Pi,color=black):
```

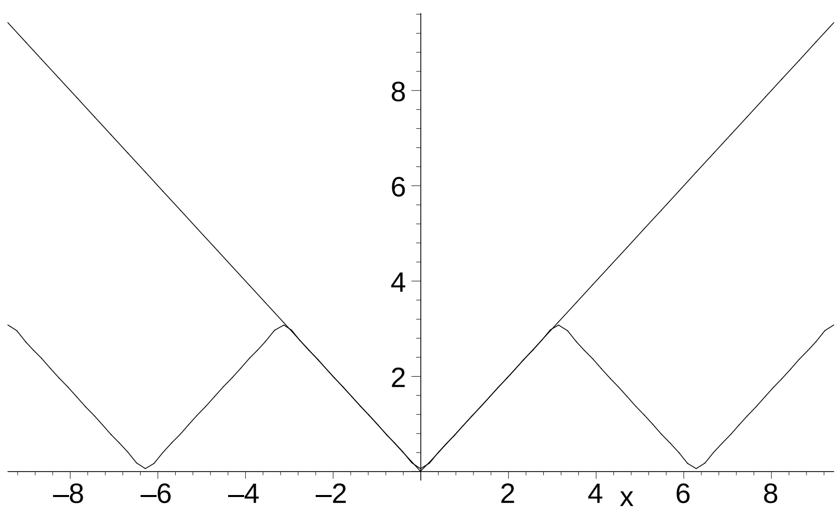
```
exact:=plot(abs(x),x=-Pi..Pi,color=black):
display({almost,exact});
```




```
>
```

On a longer interval, we see what's happening:

```
> almost:=plot(approx(x),x=-3*Pi..3*Pi,color=black):
  exact:=plot(abs(x),x=-3*Pi..3*Pi,color=black):
  display({almost,exact});
```



*What's happening is that each term in a Fourier Series has 2*Pi as a period, so the sum does as well, i.e. the Fourier series are 2*Pi periodic. So in this problem what they are really approximating is the 2*Pi-periodic extension of the absolute value function restricted to the interval [-Pi..Pi]. This is called a saw-tooth function, by the way.*

```
>
```