Eigenvalue problems

This notebook has been written in *Mathematica* by

Mark J. McCready
Professor and Chair of Chemical Engineering
University of Notre Dame
Notre Dame, IN 46556
USA

mjm@nd.edu
http://www.nd.edu/~mjm/

Version: 11/23/98

No resistance at the ends, u[0]=u[1]=0

We might start solving this problem with a homogenous boundary condition for u[0] and u[1]. However, we cannot solve all of this at once because all we will get is the trivial solution.

```
DSolve[{u''[t] + λ u[t] == 0, u[0] == 0, u[1] == 0}, u, t]
```

$\{\{u \to (0\&)\}\}$

Note that this is a more general solution (although still 0) than you get from the similar command:

```
DSolve[{u''[t] + λ u[t] == 0, u[0] == 0, u[1] == 0}, u[t], t]
```

$\{\{u(t) \to 0\}\}$

To get a solution we opt to leave off the far boundary condition. *Mathematica* is telling us that we need to adjust λ to special values to solve the problem. We first solve the ode and the u[0]=0 bondary condition.

```
ans1 = DSolve[{u''[t] + λ u[t] == 0, u[0] == 0}, u, t]
```

— *General::spell1 : Possible spelling error: new symbol name "ans1" is similar to existing symbol "ans21".*

$\{\{u \to (c_1 \sin(\sqrt{\lambda}\ \#1)\&)\}\}$

Construct the boudary condition at the far end

```
ans2= u[1] /.ans1[[1]]
```

$c_1 \sin(\sqrt{\lambda})$

We need to choose λ to make this 0. We know the answer is $\lambda = (n\pi)^2$. (Which is good because *Mathematica* does not give a useful answer.

```
eq1=ans2==0
```

$c_1 \sin(\sqrt{\lambda}) == 0$

We could find roots numerically if we had to,

```
root1=ans2/.{C[1]->1}
```

$\sin(\sqrt{\lambda})$

```
FindRoot[root1==0,{λ,9}]
```

$\{\lambda \to 9.8696\}$

```
FindRoot[root1==0,{λ,36}]
```

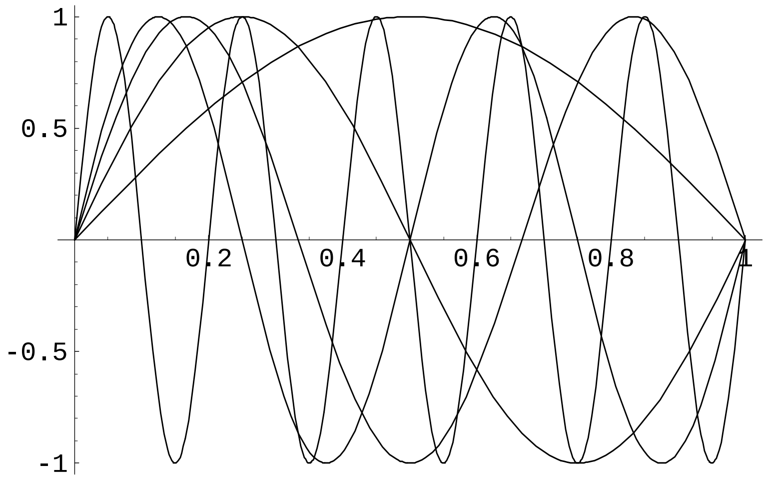
$\{\lambda \to 39.4784\}$

```
FindRoot[root1==0,{λ,84}]
```

$\{\lambda \to 88.8264\}$

Just to beat the dead horse ...

```
Plot[{Sin[Pi x], Sin[2 Pi x], Sin[3 Pi x], Sin[4 Pi x], Sin[10 Pi x]},
 {x, 0, 1}]
```



- Graphics -

Note that all of the functions "fit" on the domain -- the essential feature of an eigenvalue problem.

So we can obviously find as many values of λ as necessary. Sturm-Liouville theory guarantees that the set of eigen functions produced by finding all of the λ's will span all function space and thus any function can be expanded in terms of these (orthogonal) functions.

Check orthogonality

```
int1 = Integrate[Sin[n Pi x] Sin[m Pi x], {x, 0, 1}]
```

$$\frac{m \sin(m\pi - n\pi) + n \sin(m\pi - n\pi) - m \sin(\pi m + n\pi) + n \sin(\pi m + n\pi)}{2 (m-n) (m+n) \pi}$$

```
int2 = FullSimplify[%]
```

$$\frac{\frac{\sin((m-n)\pi)}{m-n} - \frac{\sin((m+n)\pi)}{m+n}}{2\pi}$$

This is obviously 0 if n≠m. For example,

```
int2 /. {n -> 3, m -> 8}
```

0

Otherwise,

```
int3 = Integrate[Sin[n Pi x] Sin[n Pi x], {x, 0, 1}]
```

$$\frac{1}{2} - \frac{\sin(2 n \pi)}{4 n \pi}$$

```
Table[int3, {n, 1, 10, 1}]
```

$$\left\{\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right\}$$

Thus our eigenfunctions are orthogonal and they can be easily normalized.

Now use more general boundary conditions

Now we try solving this problem with a homogenous boundary condition for u'[0] + α_1 u[0] = 0, at the front end and u'[L] + α_2u[L] =0 at the far end. However, we cannot solve all of this at once because all we will get is the trivial solution.

Thus we opt to leave off the far boundary condition. *Mathematica* is telling us that we need to adjust λ to special values to solve the problem. We first solve the ode and the u[0] boundary condition.

```
ans21 = DSolve[{u''[t] + λ u[t] == 0, -u'[0] + α₁ u[0] == 0}, u, t]
```

$$\left\{\left\{u \to \left(c_2 \cos\left(\sqrt{\lambda}\ \#1\right) + \frac{c_2 \sin\left(\sqrt{\lambda}\ \#1\right) \alpha_1}{\sqrt{\lambda}} \&\right)\right\}\right\}$$

Construct the boundary condition at the far end

```
ans22 = u'[L] + α₂ u[L] /. ans21[[1]]
```

$$-\sqrt{\lambda}\ c_2 \sin\left(L \sqrt{\lambda}\right) + c_2 \cos\left(L \sqrt{\lambda}\right) \alpha_1 + \left(c_2 \cos\left(L \sqrt{\lambda}\right) + \frac{c_2 \sin\left(L \sqrt{\lambda}\right) \alpha_1}{\sqrt{\lambda}}\right) \alpha_2$$

We need to choose λ to make this 0.

```
bc1=Expand[Expand[ans22]/Sin[L Sqrt[λ]]/C[2]]
```

$$\cot\left(L \sqrt{\lambda}\right) \alpha_1 + \frac{\alpha_2\ \alpha_1}{\sqrt{\lambda}} + \cot\left(L \sqrt{\lambda}\right) \alpha_2 - \sqrt{\lambda}$$

This takes a bit of work to make this into the nice form in V&M.

```
bc1x = Cancel[Simplify[bc1[[2]] + bc1[[3]]] / (α₁ + α₂)] +
  Cancel[Simplify[bc1[[1]] + bc1[[4]]] / (α₁ + α₂)]
```

$$\cot\left(L \sqrt{\lambda}\right) + \frac{\alpha_1\ \alpha_2 - \lambda}{\sqrt{\lambda}\ (\alpha_1 + \alpha_2)}$$

Make the substitution, λ-> s L

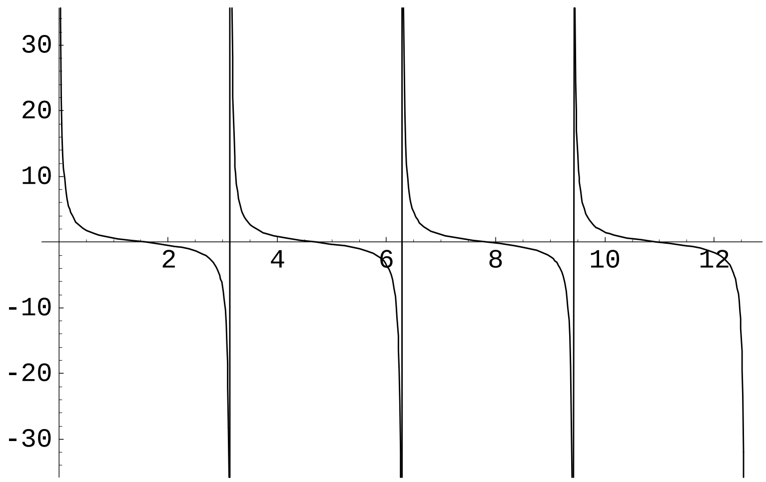
```
fofs = PowerExpand[bc1x[[1]] /. λ -> (s / L)^2]
```

$$\cot(s)$$

```
gofs = -PowerExpand[bc1x[[2]] /. λ -> (s / L)^2]
```

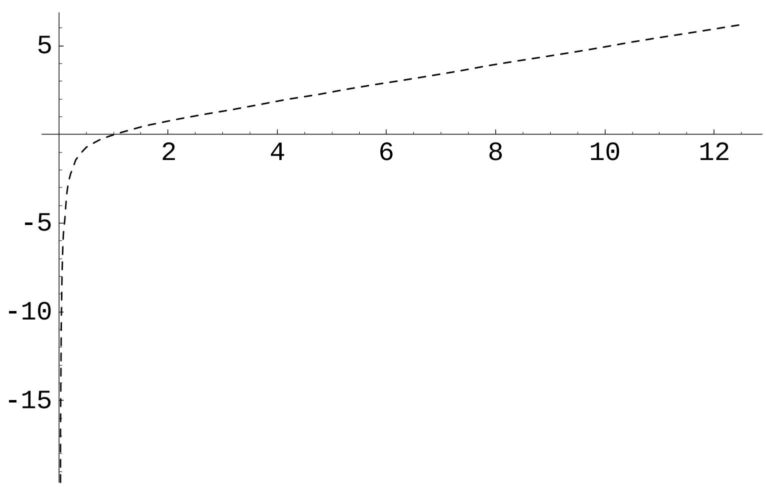
$$-\frac{L\left(\alpha_1\,\alpha_2-\frac{s^2}{L^2}\right)}{s\,(\alpha_1+\alpha_2)}$$

```
fsplot = Plot[fofs, {s, 0, 4 π}]
```



- Graphics -

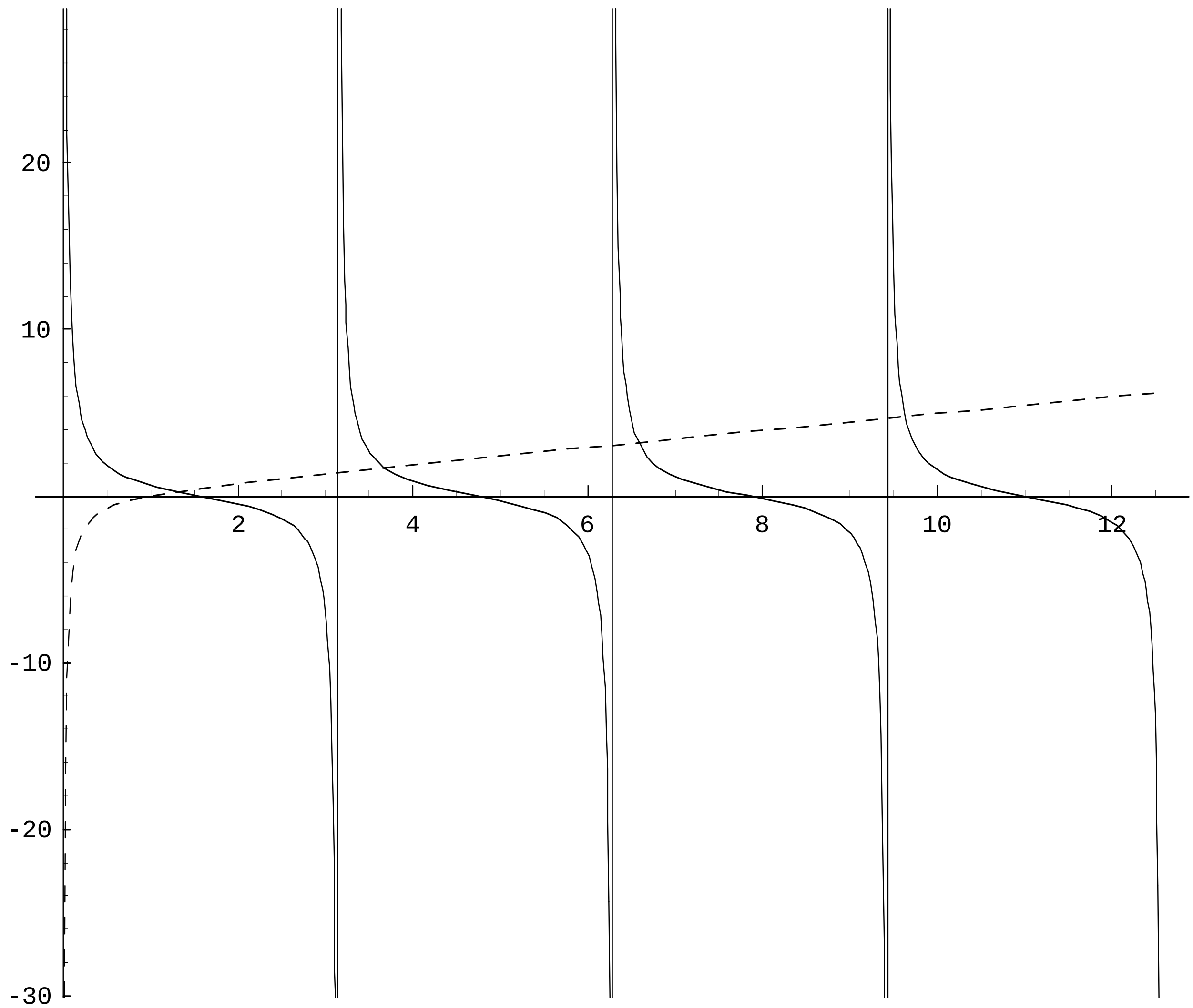
```
gsplot = Plot[gofs /. {α1 -> 1, α2 -> 1, L -> 1}, {s, 0, 4 π},
  PlotStyle -> Dashing[{.01, .01}]]
```



- Graphics -

The intersections are where the roots are,

```
Show[gsplot, fsplot]
```



- Graphics -

Let's see if we can find some of the λ's.

```
root1 = FindRoot[fofs - (gofs /. {α1 -> 1, α2 -> 1, L -> 1}) == 0, {s, 2}]
```

{$s \to 1.30654$}

```
root2 = FindRoot[fofs - (gofs /. {α1 -> 1, α2 -> 1, L -> 1}) == 0, {s, 4}]
```

{$s \to 3.67319$}

```
root3 = FindRoot[fofs - (gofs /. {α1 -> 1, α2 -> 1, L -> 1}) == 0, {s, 7}]
```

{$s \to 6.58462$}

```
root4 = FindRoot[fofs - (gofs /. {α1 -> 1, α2 -> 1, L -> 1}) == 0, {s, 10}]
```

$\{s \to 9.63168\}$

```
func = (u[x] /. ans21) /. {α1 -> 1, α2 -> 1, L -> 1, C[2] -> 1}
```

$\left\{\cos\left(x\sqrt{\lambda}\right) + \frac{\sin\left(x\sqrt{\lambda}\right)}{\sqrt{\lambda}}\right\}$

```
(func /. √λ -> s) /. root1
```

$\left\{\cos(1.30654\, x) + \frac{\sin(1.30654\, x)}{\sqrt{\lambda}}\right\}$

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root1, {x, 0, 1}]
```

You see that the "fit" is of a different form for the more complex boundary conditions

Now verify orthogonality.

```
Integrate[((func /. {√λ -> s, 1/√λ -> 1/s}) /. root4) *
  ((func /. {√λ -> s, 1/√λ -> 1/s}) /. root1)
, {x, 0, 1}]
```

$\{-5.93384\times 10^{-10} + 0.\, i\}$

```
Integrate[((func /. {√λ -> s, 1/√λ -> 1/s}) /. root1) *
  ((func /. {√λ -> s, 1/√λ -> 1/s}) /. root1)
, {x, 0, 1}]
```

$\{1.37871 + 0.\, i\}$

```
Integrate[((func /. {√λ -> s, 1/√λ -> 1/s}) /. root4) *
  ((func /. {√λ -> s, 1/√λ -> 1/s}) /. root3)
, {x, 0, 1}]
```

$\{9.50542\times 10^{-10} + 0.\, i\}$

```
Integrate[((func /. {√λ -> s, 1/√λ -> 1/s}) /. root3) *
  ((func /. {√λ -> s, 1/√λ -> 1/s}) /. root3)
, {x, 0, 1}]
```

$\{0.534596 + 0.\, i\}$

```
Integrate[((func /. {√λ -> s, 1/√λ -> 1/s}) /. root4) *
  ((func /. {√λ -> s, 1/√λ -> 1/s}) /. root4)
, {x, 0, 1}]
```

{0.516169 + 0. 𝕚}

```
Integrate[ (func[[1]] /. {λ -> λ₁}) *  (func[[1]] /. {λ -> λ₂}),
 {x, 0, 1}]
```

— *Integrate::gener : Unable to check convergence*

$$\left(\left(\cos(\sqrt{\lambda_1}) + \frac{\sin(\sqrt{\lambda_1})}{\sqrt{\lambda_1}}\right)\left(\cos(\sqrt{\lambda_2}) + \frac{\sin(\sqrt{\lambda_2})}{\sqrt{\lambda_2}}\right)\left(\sqrt{\lambda_1}\,\lambda_2 \sin(\sqrt{\lambda_1} - \sqrt{\lambda_2}) + \sqrt{\lambda_1}\,\sin(\sqrt{\lambda_1} - \sqrt{\lambda_2}) + \lambda_1 \sqrt{\lambda_2}\,\sin(\sqrt{\lambda_1} - \sqrt{\lambda_2}) + \sqrt{\lambda_2}\,\sin(\sqrt{\lambda_1} - \sqrt{\lambda_2}) + \cos(\sqrt{\lambda_1} - \sqrt{\lambda_2})\lambda_1 - \cos(\sqrt{\lambda_1} + \sqrt{\lambda_2})\lambda_1 - \cos(\sqrt{\lambda_1} - \sqrt{\lambda_2})\lambda_2 + \cos(\sqrt{\lambda_1} + \sqrt{\lambda_2})\lambda_2 - \sin(\sqrt{\lambda_1} + \sqrt{\lambda_2})\sqrt{\lambda_1}\,\lambda_2 - \sin(\sqrt{\lambda_1} + \sqrt{\lambda_2})\sqrt{\lambda_1} + \sin(\sqrt{\lambda_1} + \sqrt{\lambda_2})\sqrt{\lambda_2} + \sin(\sqrt{\lambda_1} + \sqrt{\lambda_2})\lambda_1 \sqrt{\lambda_2}\right)\right) \Big/ \left(2\left(\sqrt{\lambda_1}\,\cos(\sqrt{\lambda_1}) + \sin(\sqrt{\lambda_1})\right)\left(\sqrt{\lambda_1} - \sqrt{\lambda_2}\right)\left(\sqrt{\lambda_1} + \sqrt{\lambda_2}\right)\left(\sqrt{\lambda_2}\,\cos(\sqrt{\lambda_2}) + \sin(\sqrt{\lambda_2})\right)\right)$$

```
FullSimplify[%]
```

$$\frac{\frac{\sin(\sqrt{\lambda_1})\sqrt{\lambda_1}\,(\sqrt{\lambda_2}\,\cos(\sqrt{\lambda_2})+\sin(\sqrt{\lambda_2}))}{\sqrt{\lambda_2}} + \frac{\sin(\sqrt{\lambda_1})(\cos(\sqrt{\lambda_2})-\sin(\sqrt{\lambda_2})\sqrt{\lambda_2})}{\sqrt{\lambda_1}} - \frac{\cos(\sqrt{\lambda_1})\sin(\sqrt{\lambda_2})(\lambda_2+1)}{\sqrt{\lambda_2}}}{\lambda_1 - \lambda_2}$$

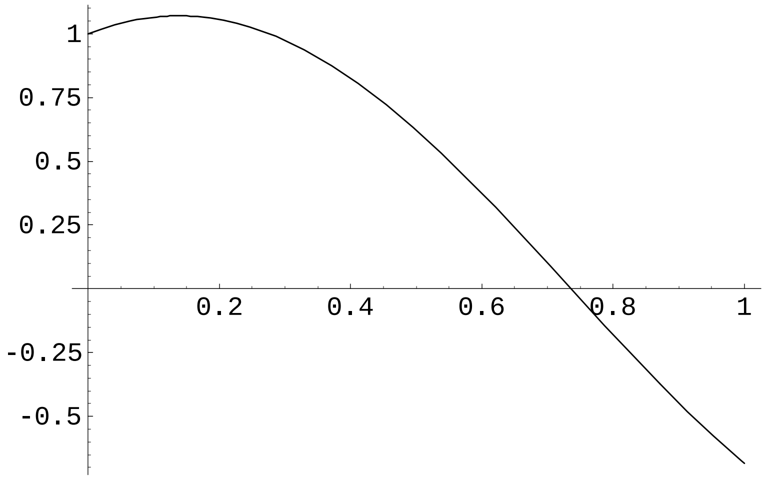
You can verify how the normalization should go.

Now decrease the external resistance by increasing α.

```
gsplot2 = Plot[gofs /. {α₁ -> 10, α₂ -> 10, L -> 1}, {s, 0, 4 π},
  PlotStyle -> Dashing[{.01, .01}]]
```

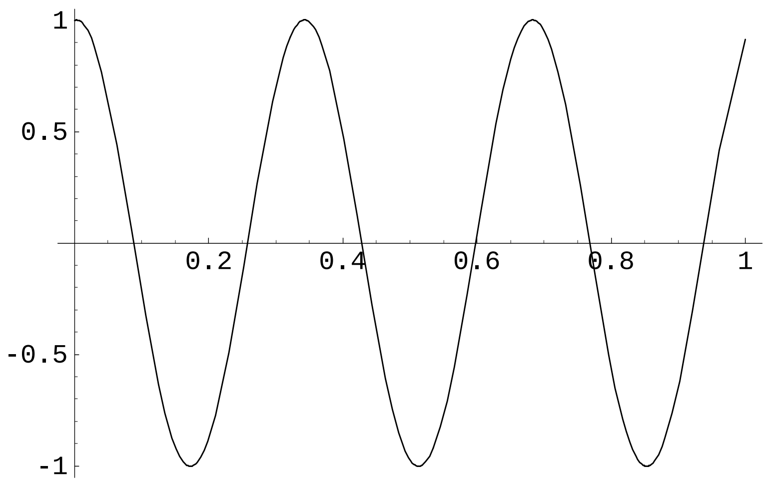
We should find that we are approaching the no resistance limit,

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root1x, {x, 0, 1}]
```



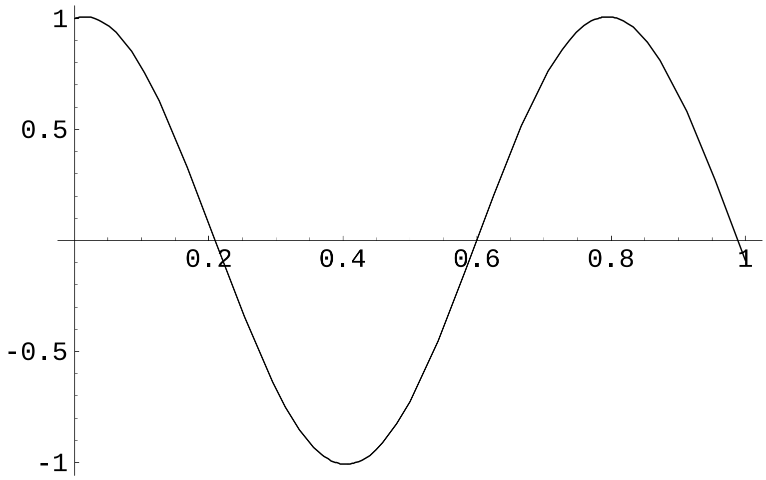
- Graphics -

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root2x, {x, 0, 1}]
```



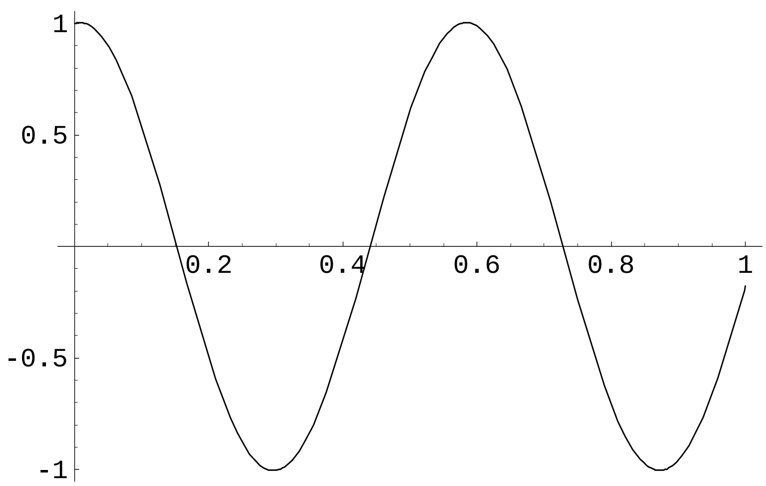
- Graphics -

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root3x, {x, 0, 1}]
```



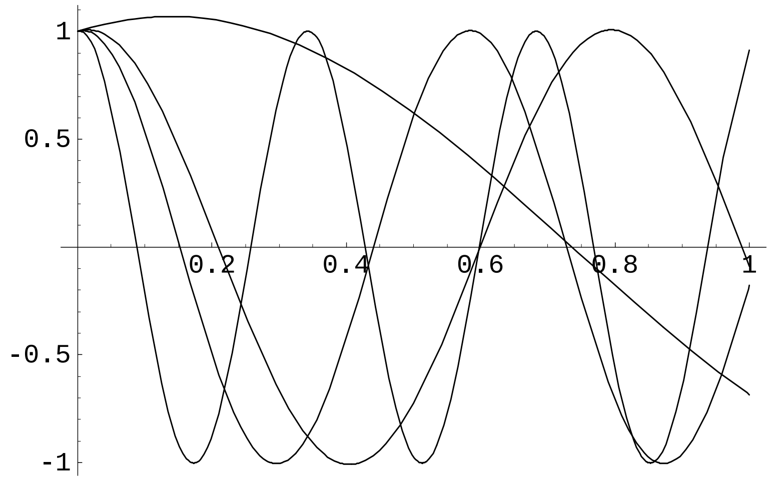
- Graphics -

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root4x, {x, 0, 1}]
```



- Graphics -

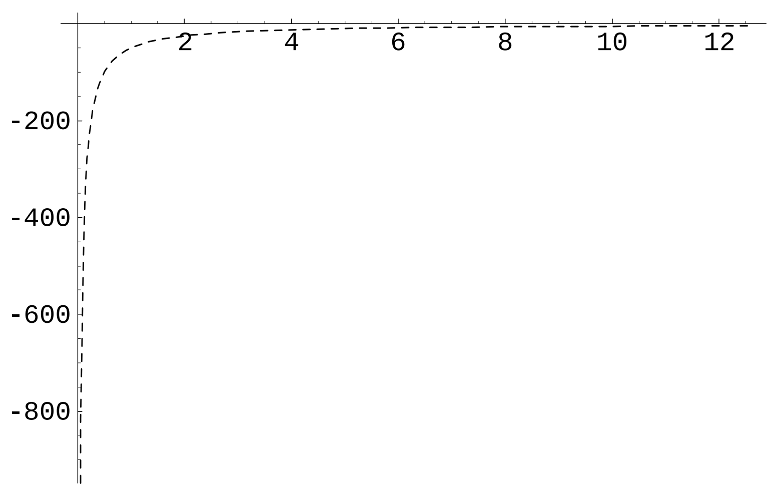
```
Show[%, %%, %%%, %%%%]
```



- Graphics -

Now decrease the external resistance by increasing α.

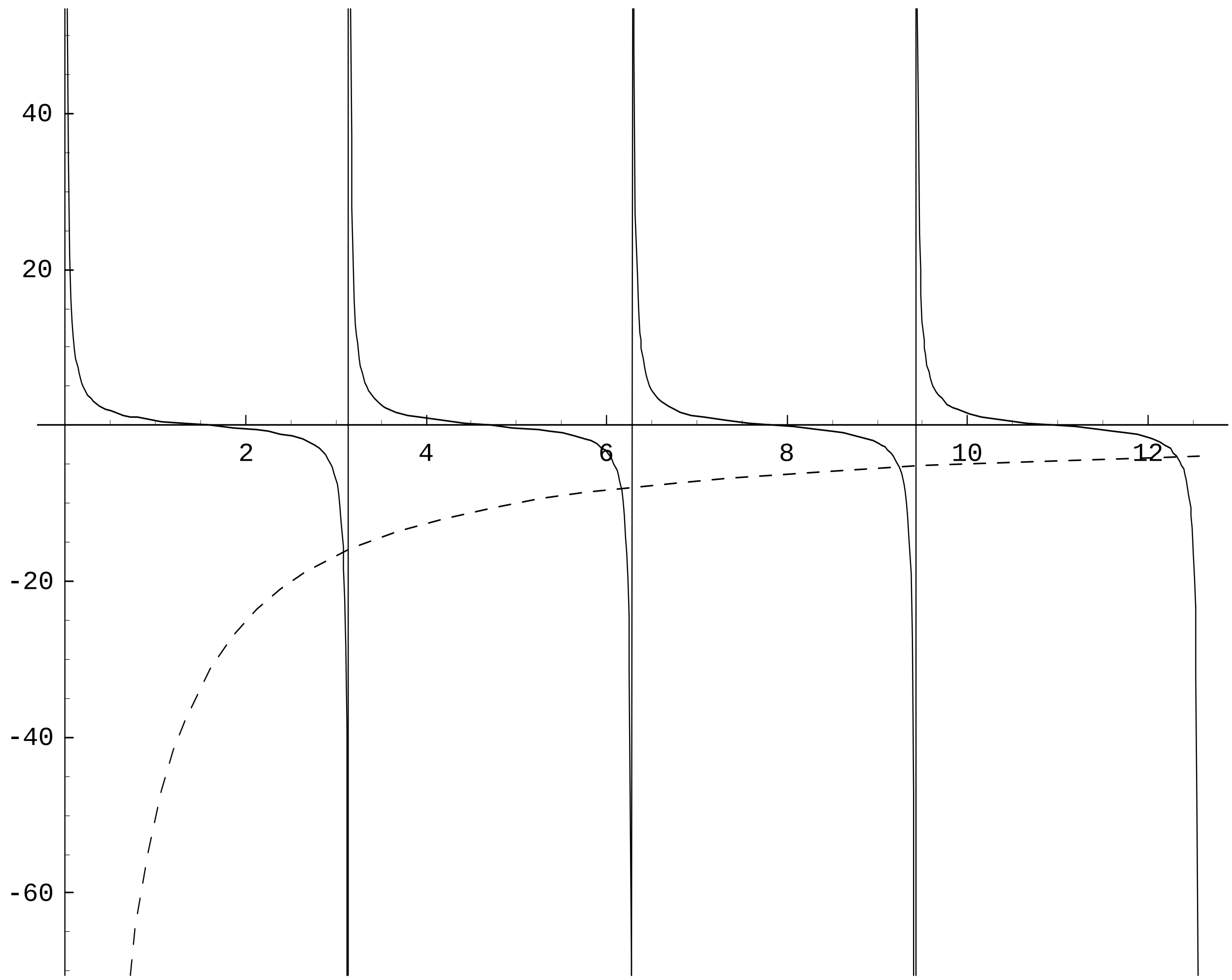
```
gsplot4 = Plot[gofs /. {α1 -> 100, α2 -> 100, L -> 1}, {s, 0, 4 π},
  PlotStyle -> Dashing[{.01, .01}]]
```



- Graphics -

The intersections are where the roots are,

```
Show[gsplot4, fsplot]
```



- Graphics -

Let's see if we can find some of the λ's.

```
root1z =
 FindRoot[fofs - (gofs /. {α1 -> 100, α2 -> 100, L -> 1}) == 0, {s, 3}]
```

— *General::spell : Possible spelling error: new symbol name "root1z" is similar to existing symbols {root1, root1x, root1y}.*

$\{s \to 3.08001\}$

```
root2z =
 FindRoot[fofs - (gofs /. {α1 -> 100, α2 -> 100, L -> 1}) == 0, {s, 6.2}]
```

$\{s \to 6.16014\}$

```
root3z =
 FindRoot[fofs - (gofs /. {α1 -> 100, α2 -> 100, L -> 1}) == 0, {s, 9}]
```

$\{s \to 9.24049\}$

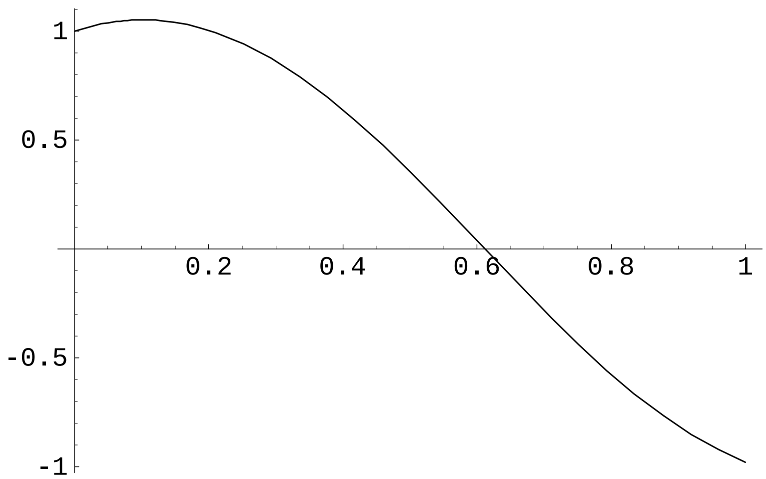
```
root4z =
 FindRoot[fofs - (gofs /. {α1 -> 100, α2 -> 100, L -> 1}) == 0, {s, 12}]
```

— *General::spell : Possible spelling error: new symbol name "root4z" is similar to existing symbols {root4, root4x, root4y}.*

$\{s \to 12.3212\}$

We should find that we are approaching the no resistance limit,

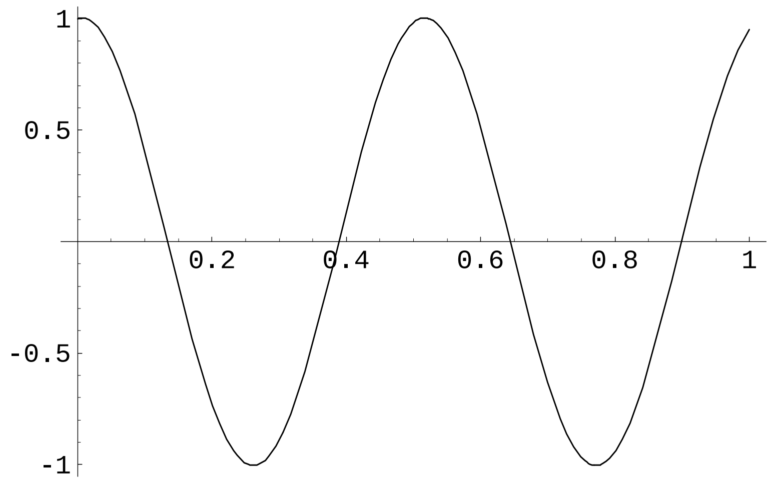
```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root1z, {x, 0, 1}]
```



- Graphics -

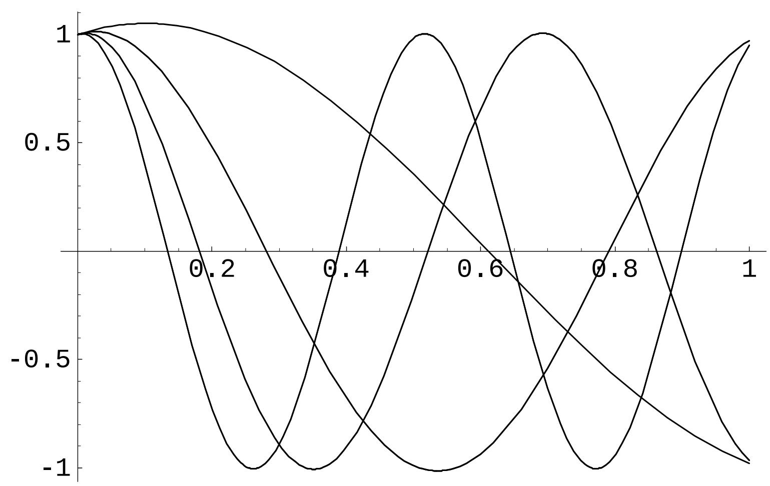
```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root2z, {x, 0, 1}]
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root3z, {x, 0, 1}]
```

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root4z, {x, 0, 1}]
```



- Graphics -

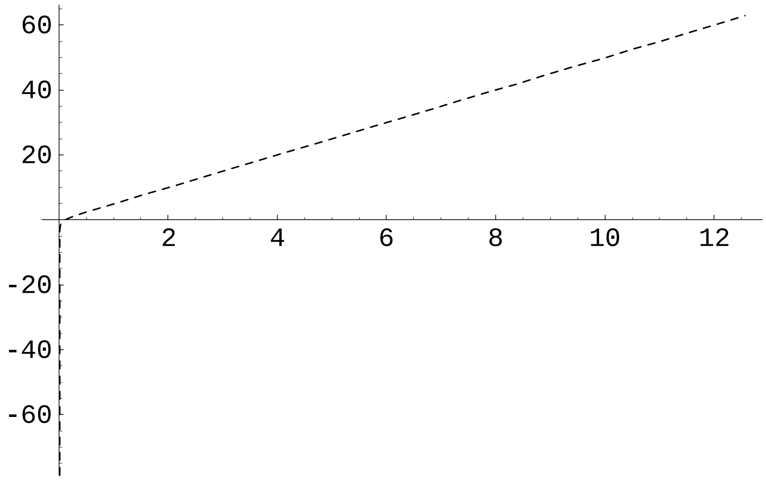
```
Show[%, %%, %%%, %%%%]
```



- Graphics -

Now increase the external resistance by decreasing α.

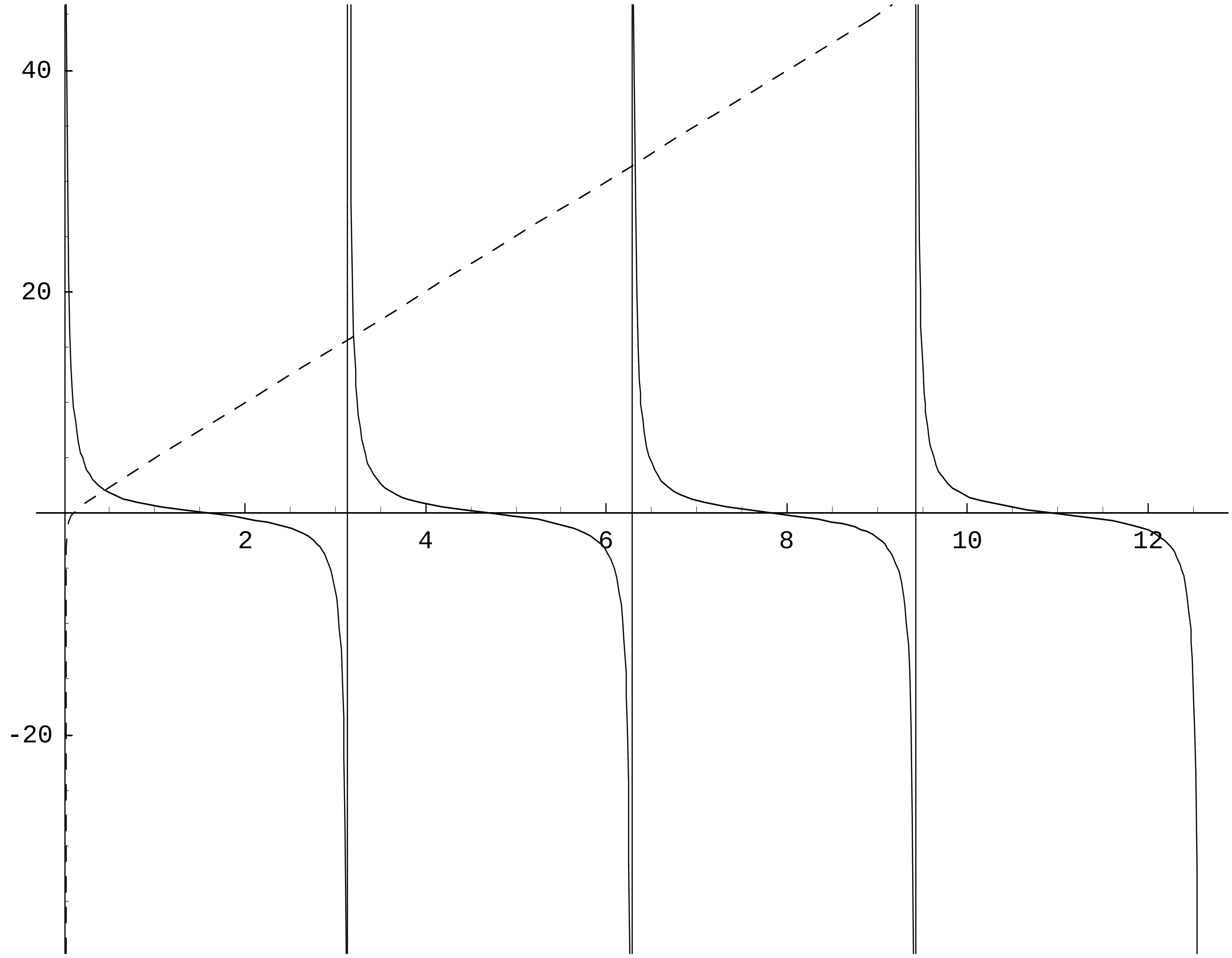
```
gsplot3 = Plot[gofs /. {α1 -> .1, α2 -> .1, L -> 1}, {s, 0, 4 π},
  PlotStyle -> Dashing[{.01, .01}]]
```



- Graphics -

The intersections are where the roots are,

```
Show[gsplot3, fsplot]
```



- Graphics -

Let's see if we can find some of the λ's.

```
root1y =
 FindRoot[fofs - (gofs /. {α1 -> .1, α2 -> .1, L -> 1}) == 0, {s, 2}]
```

{$s \to 0.443521$}

```
root2y =
 FindRoot[fofs - (gofs /. {α1 -> .1, α2 -> .10, L -> 1}) == 0, {s, 3.2}]
```

{$s \to 3.20399$}

```
root3y =
 FindRoot[fofs - (gofs /. {α1 -> .10, α2 -> .10, L -> 1}) == 0, {s, 6.5}]
```

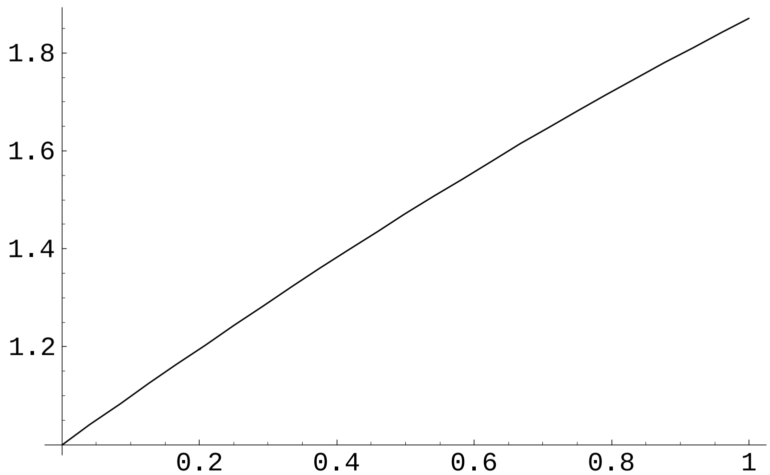
$\{s \to 6.31485\}$

```
root4y =
 FindRoot[fofs - (gofs /. {α1 -> .10, α2 -> .10, L -> 1}) == 0, {s, 9.5}]
```

$\{s \to 9.44595\}$

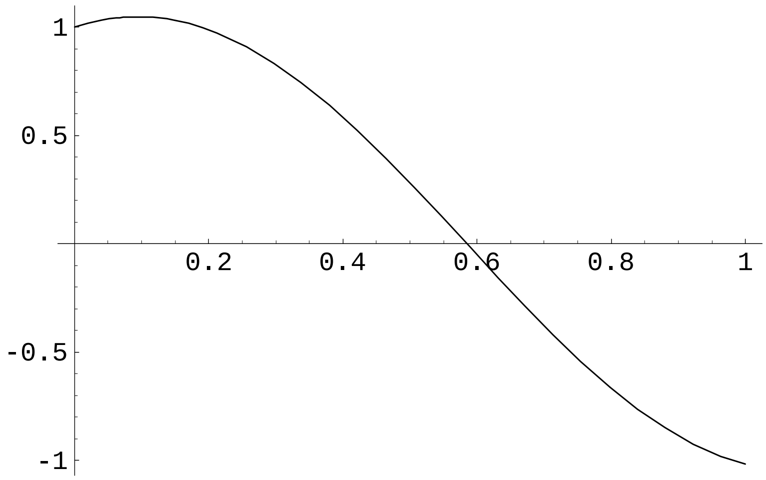
We should find that we are approaching the no resistance limit,

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root1y, {x, 0, 1}]
```



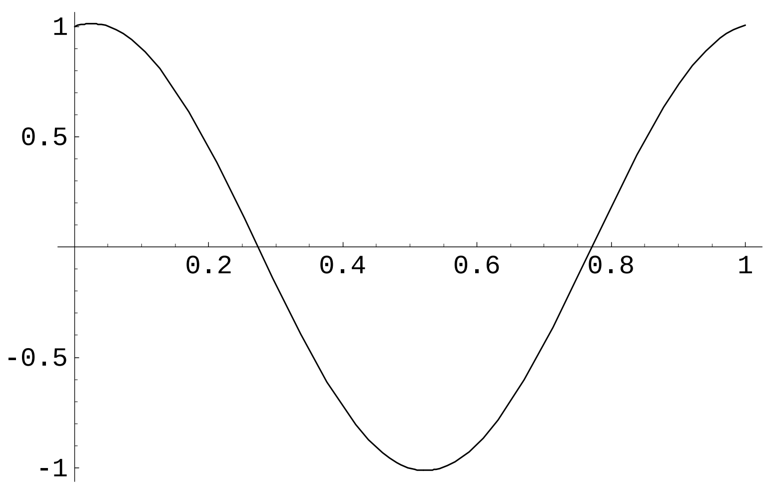
- Graphics -

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root2y, {x, 0, 1}]
```



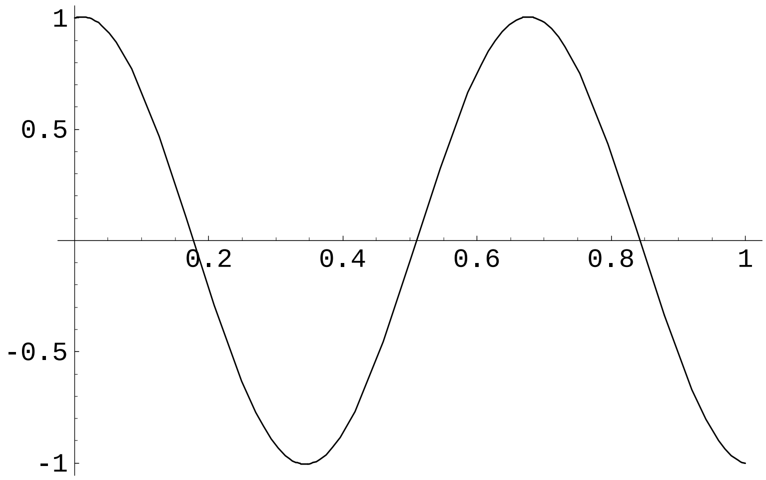
- Graphics -

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root3y, {x, 0, 1}]
```



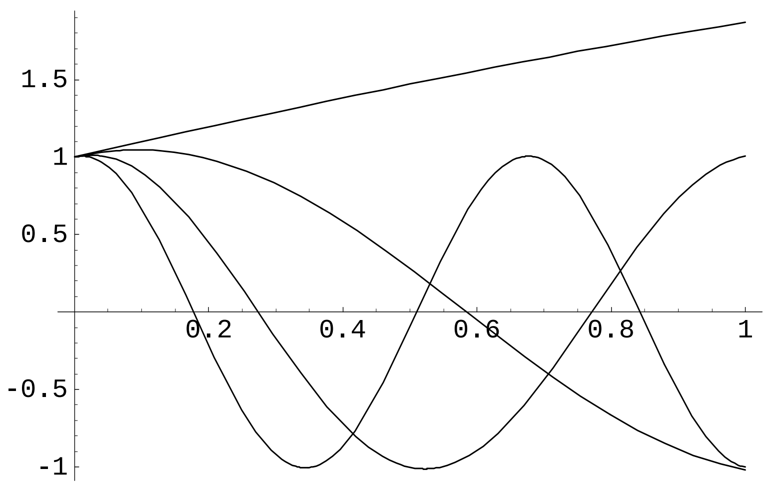
- Graphics -

```
Plot[(func /. {√λ -> s, 1/√λ -> 1/s}) /. root4y, {x, 0, 1}]
```



- Graphics -

```
Show[%, %%, %%%, %%%%]
```



- Graphics -