Problem 7.6d Joel Shapiro

As sigma_1 sets the scale for sigma we can choose it to be 1, and use tau for ct.
gamma is a reserved word, so I use gam for it. Let xp and yp be the sigma derivatives of x and y. So

```
> xp:= (tau,sigma) -> cos(gam * sin(Pi*tau)*sin(Pi*sigma))* cos(gam * cos(Pi*tau)*cos(Pi*sigma));
```

$$xp := (\tau, \sigma) \to \cos(gam \sin(\pi \tau) \sin(\pi \sigma)) \cos(gam \cos(\pi \tau) \cos(\pi \sigma))$$

```
> yp:=(tau,sigma) -> cos(gam * sin(Pi*tau)*sin(Pi*sigma))*sin (gam * cos(Pi*tau)*cos(Pi*sigma));
```

$$yp := (\tau, \sigma) \to \cos(gam \sin(\pi \tau) \sin(\pi \sigma)) \sin(gam \cos(\pi \tau) \cos(\pi \sigma))$$

The sigma=0 endpoint is at x=y=0, so x and y are just the integral of xp and yp from 0 to sigma

```
x:= (tau,sigma) -> int(xp(tau,s),s=0..sigma);
```

$$x := (\tau, \sigma) \to \int_0^\sigma \mathrm{xp}(\tau, s)\, ds$$

```
> y:= (tau,sigma) -> int(yp(tau,s),s=0..sigma);
```

$$y := (\tau, \sigma) \to \int_0^\sigma \mathrm{yp}(\tau, s)\, ds$$

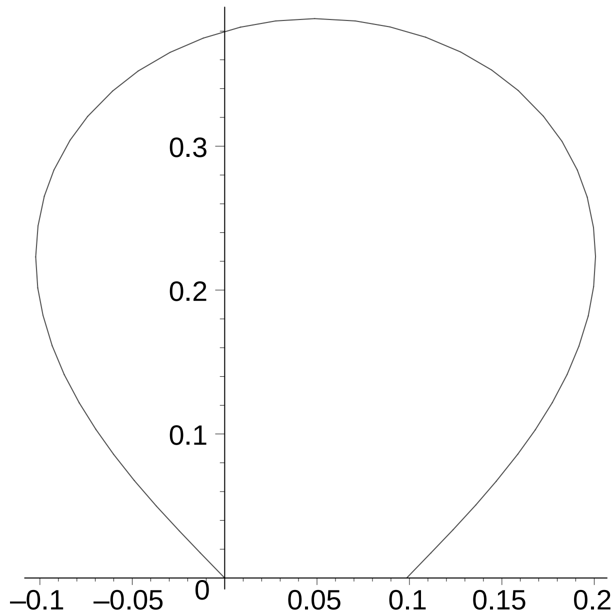
The book asks us to use

```
> gam := Pi/sqrt(2);
```

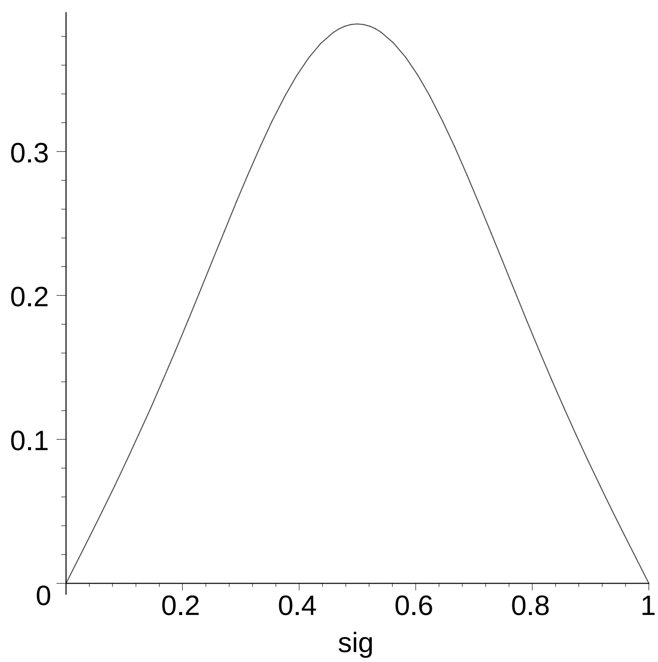
$$gam := \frac{\pi \sqrt{2}}{2}$$

First let's look at t=0:

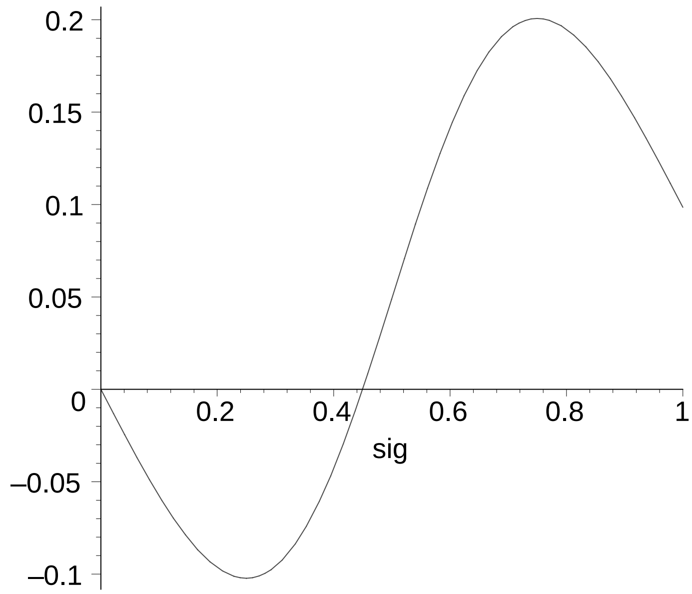
```
> plot([x(0,sig),y(0,sig),sig=0..1]);
```




```
> plot(y(0,sig),sig=0..1);
```




```
> plot(x(0,sig),sig=0..1);
```



Let’s evaluate a for this gamma, with the scale set by sigma_1=1:

```
> evalf(x(0,1));
```

0.09847494076

sigma_1 sets the scale, all that is really dtermined is a/sigma_1. So if we want a=1, we need to set sigma_1 to the inverse of this:

```
> 1/%;
```

10.15486775

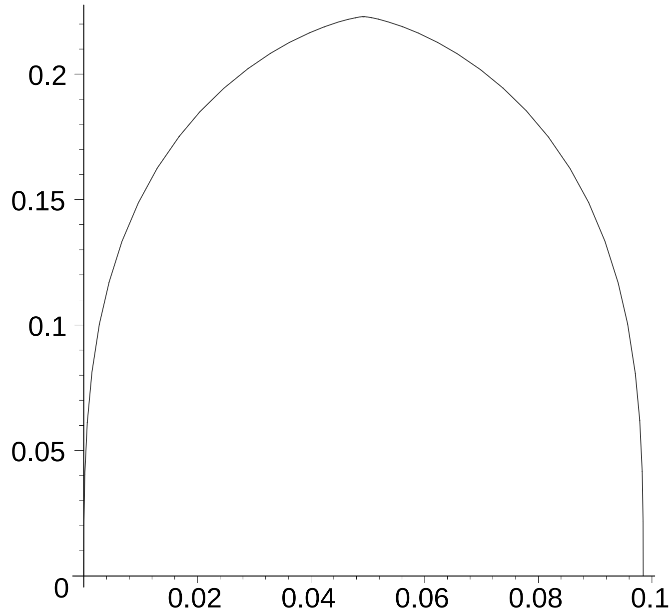
which agrees with the book. Just to check on the bessel function J_0(gam)

```
> evalf(BesselJ(0,gam));
```

0.09847494081

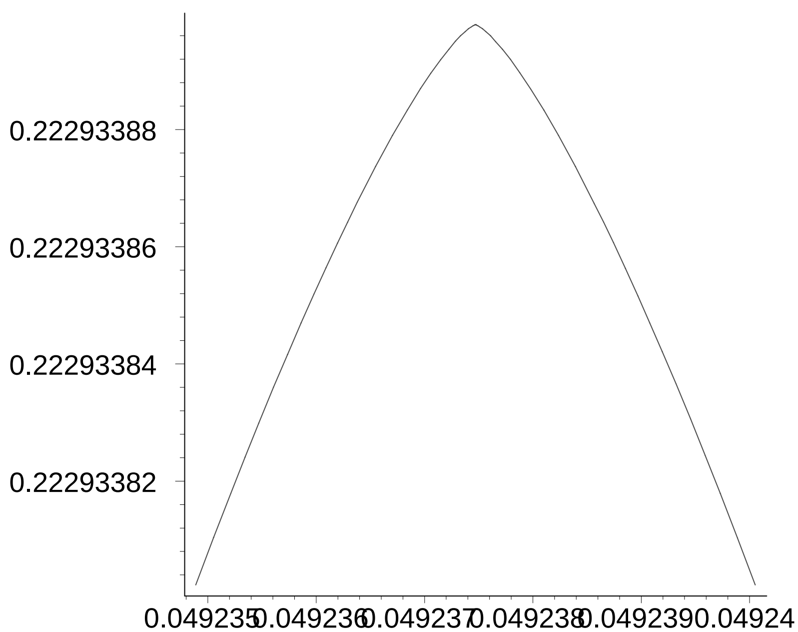
Now lets look at the string at time sigma_1/4:

```
> plot([x(1/4,sig),y(1/4,sig),sig=0..1]);
```



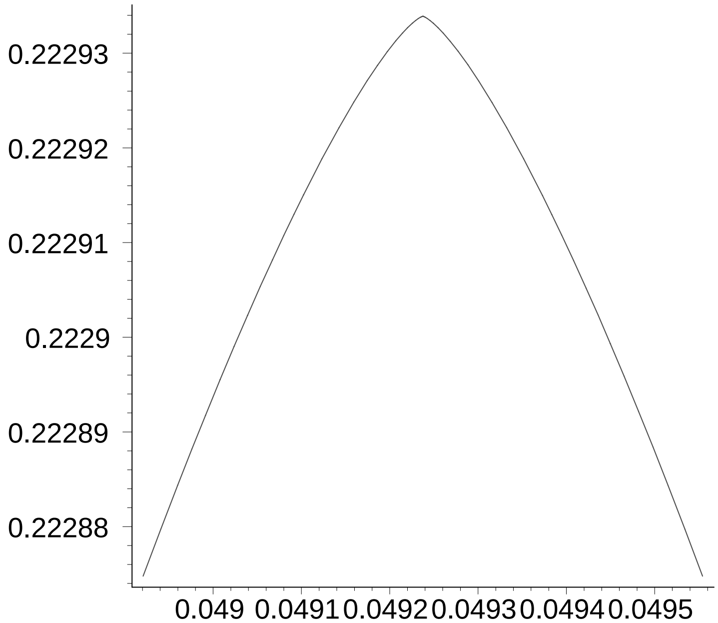
Was that a kink? Lets blow up the region, which looks like it would be at sigma=1/2:

```
plot([x(1/4,sig),y(1/4,sig),sig=0.49..0.51]);
```



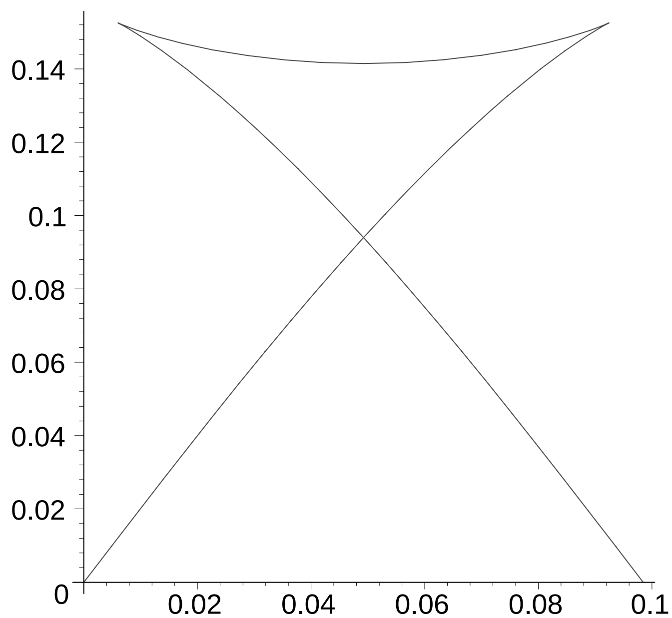
That looks kinky, but also shows numerical method imprecision on the screen. (For some reason, the printed version seems more accurate). Lets try again:

```
plot([x(1/4,sig),y(1/4,sig),sig=0.45..0.55]);
```



I guess that’s a kink. No question here, though, at t=sigma_1/3:

```
plot([x(1/3,sig),y(1/3,sig),sig=0..1]);
```




```
>
```