Examples

January 18, 2026

[17]:
```
include("notebook_init.jl");
```

Lei. I have modified alg 1 and alg 2 to keep going even if the gradient norm increases. This makes a significant difference for Example 2. We will need to put the updated figures into the paper and **make the notation consistent**. I think the results for Example 2 look ok. I am troubled that we can only drive the relative gradient norm to about .5. Is that a problem?

This notebook generates the figures in the paper *Complexity of Projected Gradient Methods for Strongly Convex Optimization with H{"o}lder Continuous Gradient Terms* by X. Chen, C. T. Kelley, and L. Wang

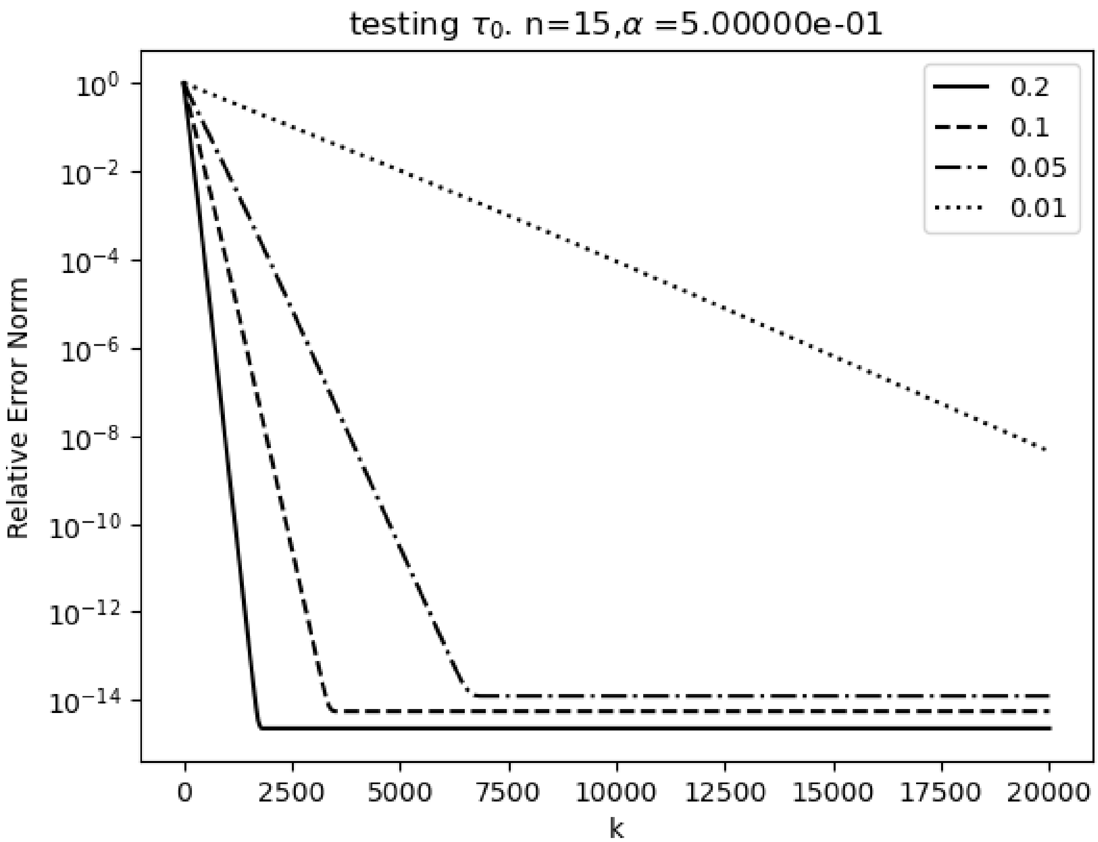
Example 1

Generate Figures 1(a), 1(b), 2(a), and 2(b). Theses figures are for Algorithm 1. Figures 1(a) and 2(a) compare various stepsizes $\tau = \tau_0 h^2$ which are consistent with the CFL condition. Figures 1(b) and 2(b) examine values of the exponent p.

The files for building the figures are in /**src**/**Figures**. The code is **Figures_Alg1.jl** and the functions **Figure1_2a** and **Figure1_2b**. The functions take the dimension as an argument.

[18]:
```
Figure1_2a(15);
```

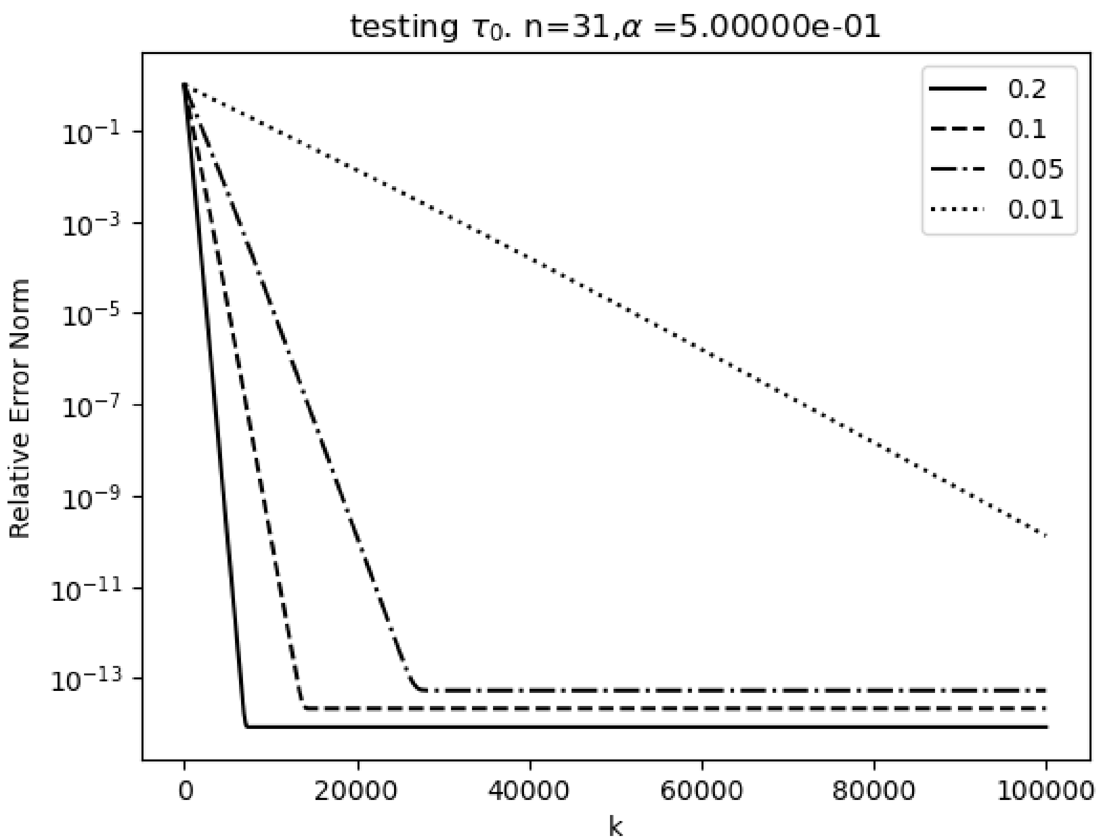
```
testing $\tau_0$. n=15,$\alpha$ =5.00000e-01
```



We run this again with a problem size of 31x31. This requiers smaller stepsizes ans the Lipschitz constant increases by a factor of four.

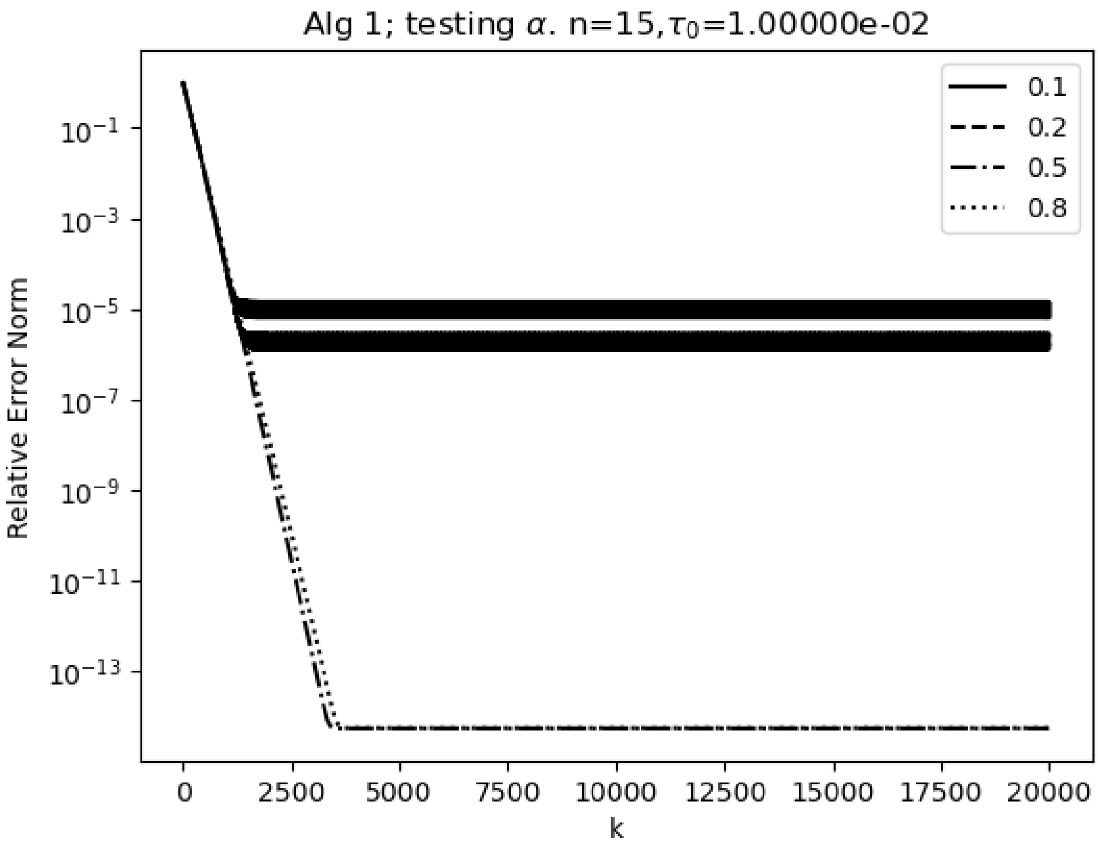
[19]:
```
Figure1_2a(31);
```

```
testing $\tau_0$. n=31,$\alpha$ =5.00000e-01
```



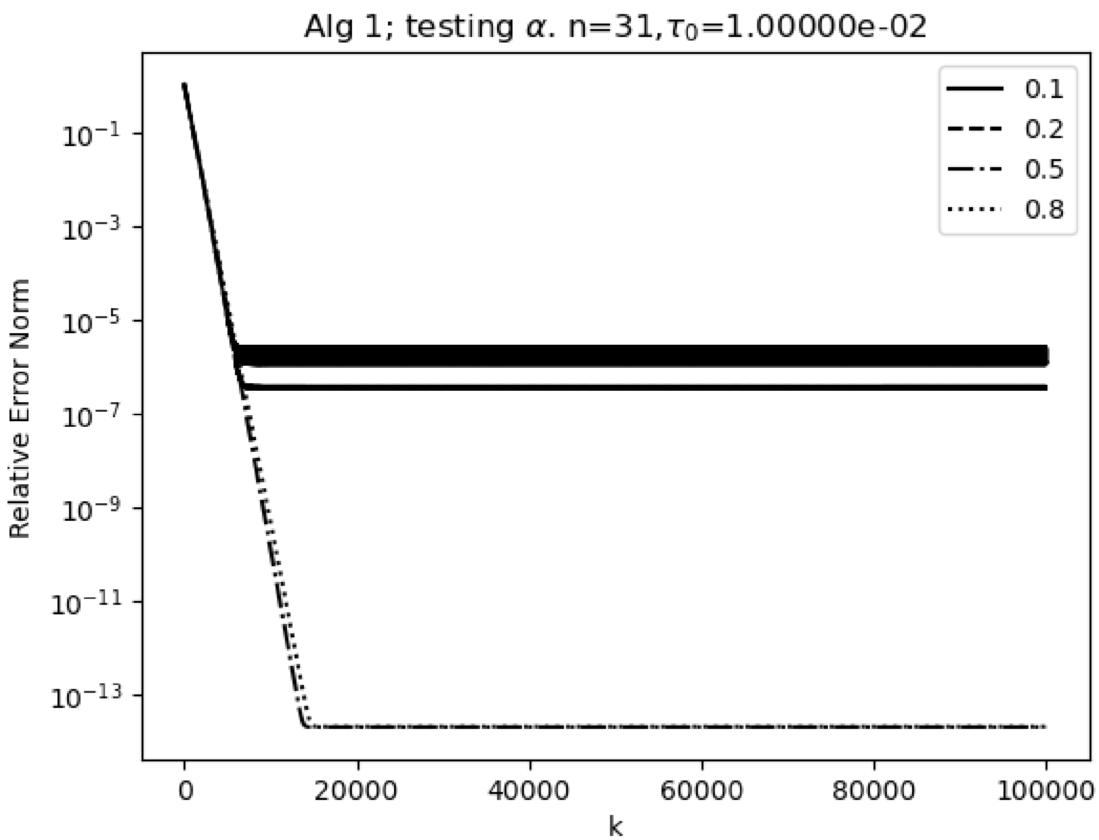
Now we compare the effects of changing the exponent p. Here we can see the effects of the change I made in Alg 1 by letting the gradient norm increase without terminating the iteration.

[20]:
```
Figure1_2b(15);
```



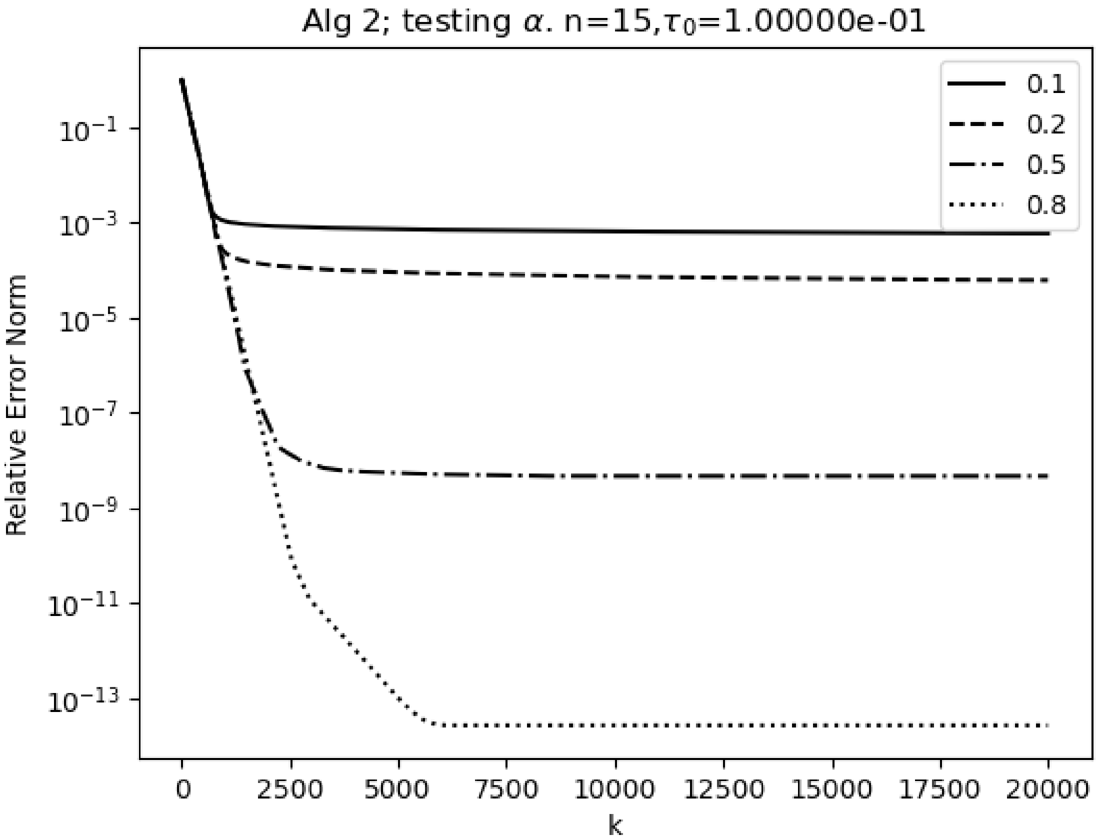
And finally repeat the computation for a 31x31 grid. This will complete the computations for Example 1 + Algorithm 1.

[21]:
```
Figure1_2b(31);
```



If we are no longer considering the line search, we should remove the discussion of Alg 2. The next example is Figure 3, which tests Algorithm 2. We start with $\tau_0 = 1$ and let the line search work.

[22]:
```
Figure3(15);
```

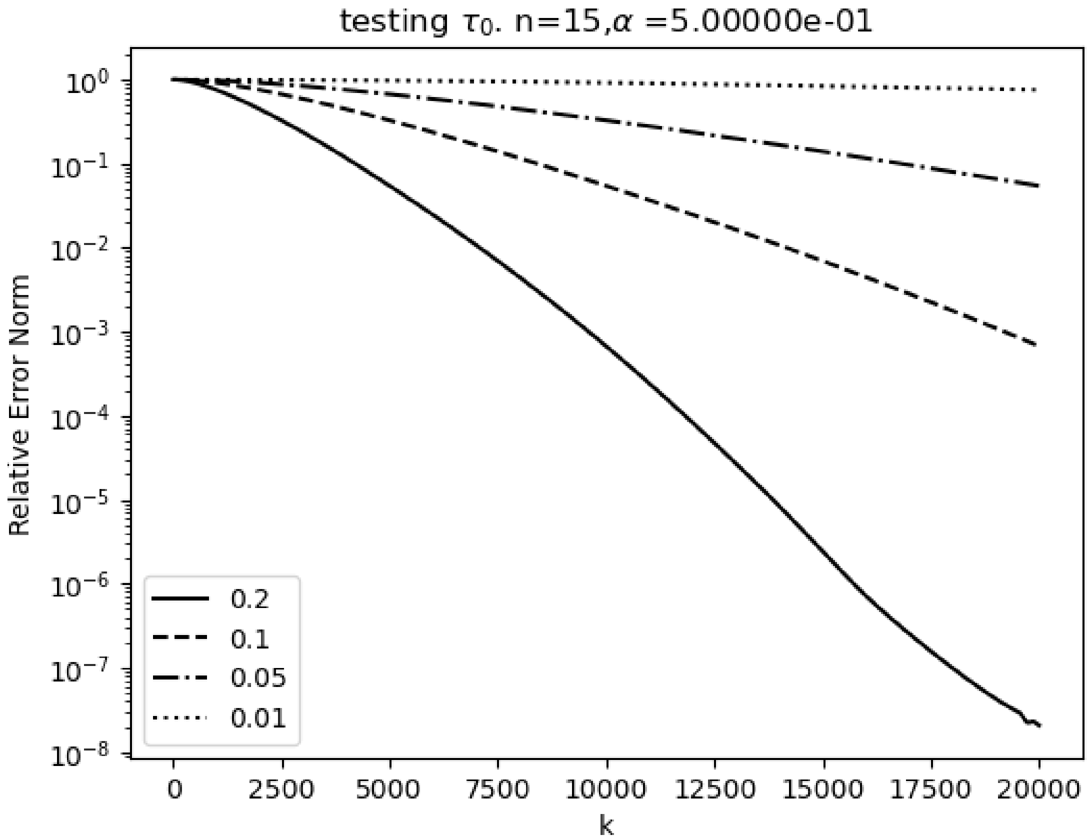


The advantage of the line search is that one does not have to manually adjust τ_0.

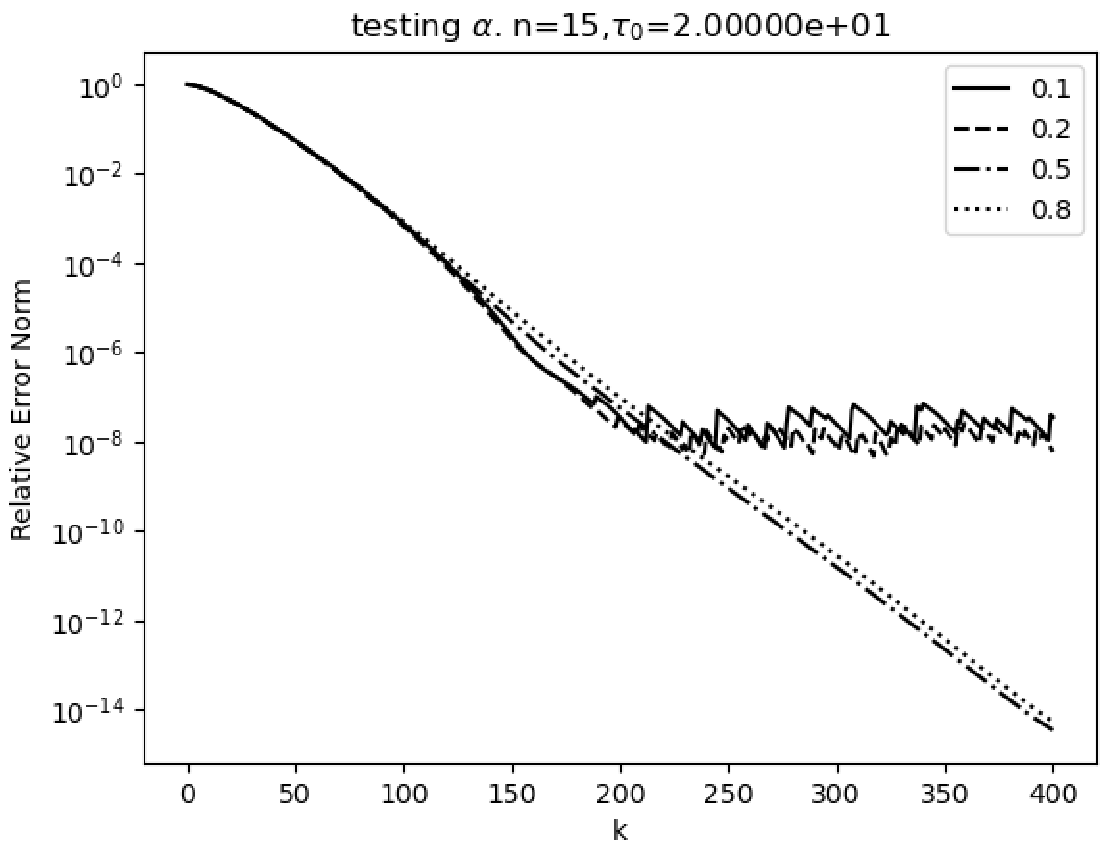
The results for Algoirthm 3 are in Figures 4 and 5. We set $\nu = \tau_0 h^2$ in these examples and will need to modify that to use the estimate in Remark 4.2. In the first two figures 4(a) and 4(b) use use the values of τ_0 we use in Figure 1.

[23]:
```
Figure4_5a(15);
```

```
testing $\tau_0$. n=15,$\alpha$ =5.00000e-01
```



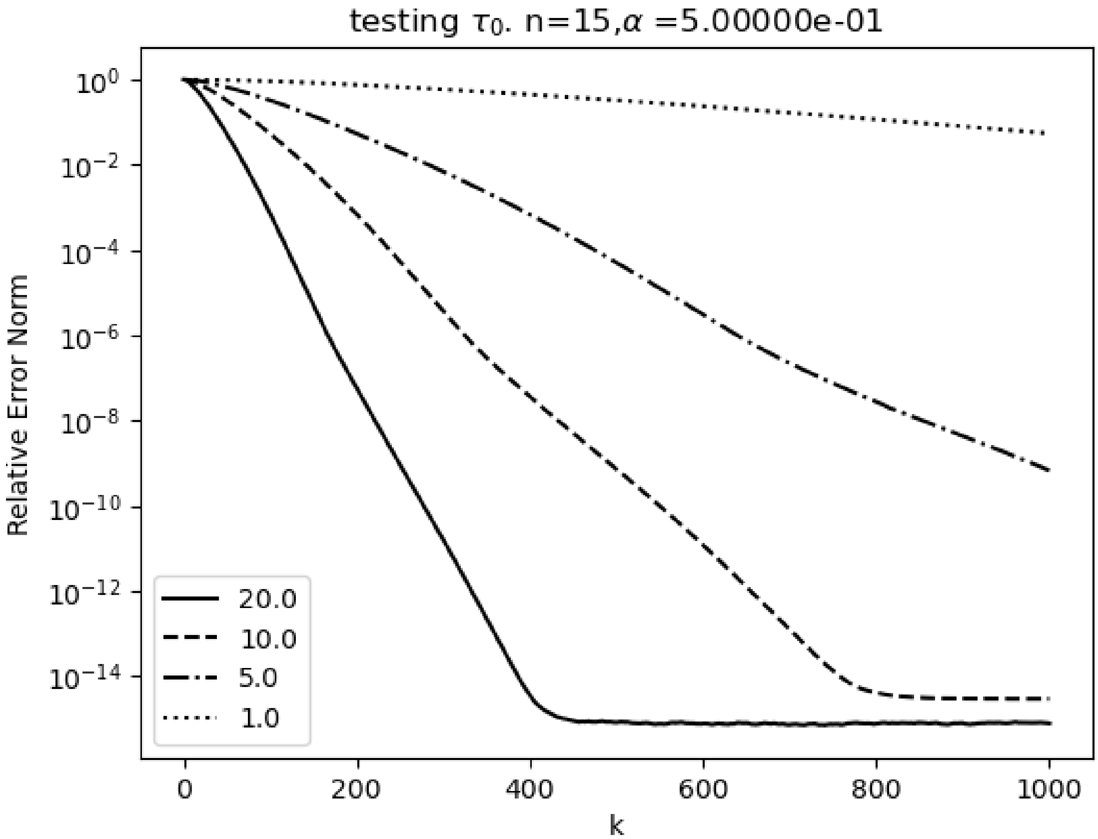

```
[24]: Figure4_5b(15);
```



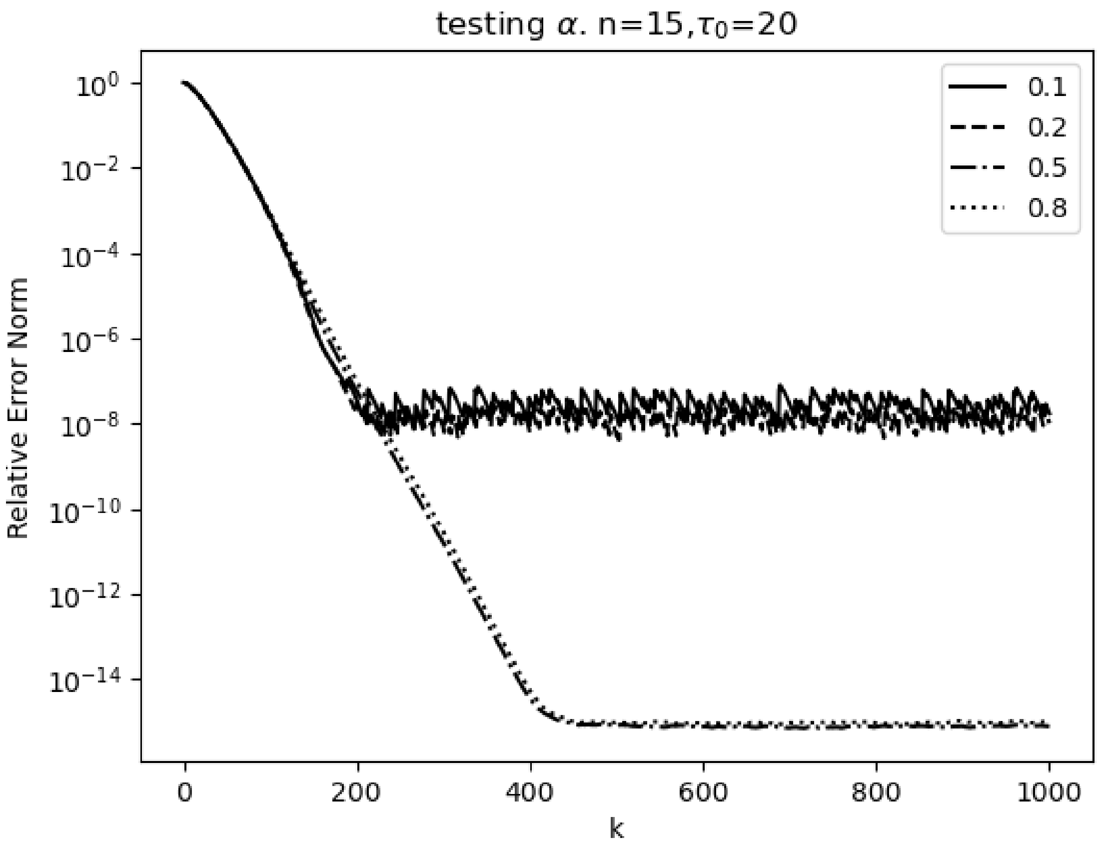
Now we use the larger values of τ_0.

[25]:
```
Figure4_5a(15; maxit=1000, tauvec=[20.0, 10.0, 5.0, 1.0]);
```

```
testing $\tau_0$. n=15,$\alpha$ =5.00000e-01
```

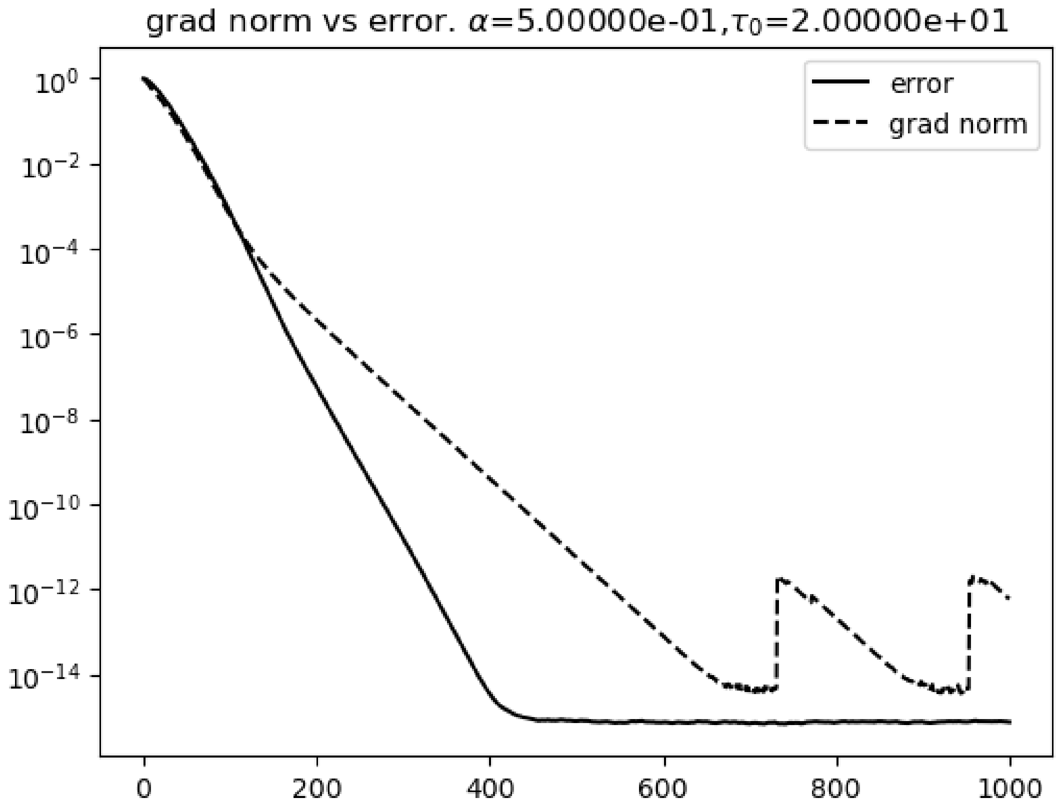


[26]:
```
Figure4_5b(15; maxit=1000, tau0=20);
```

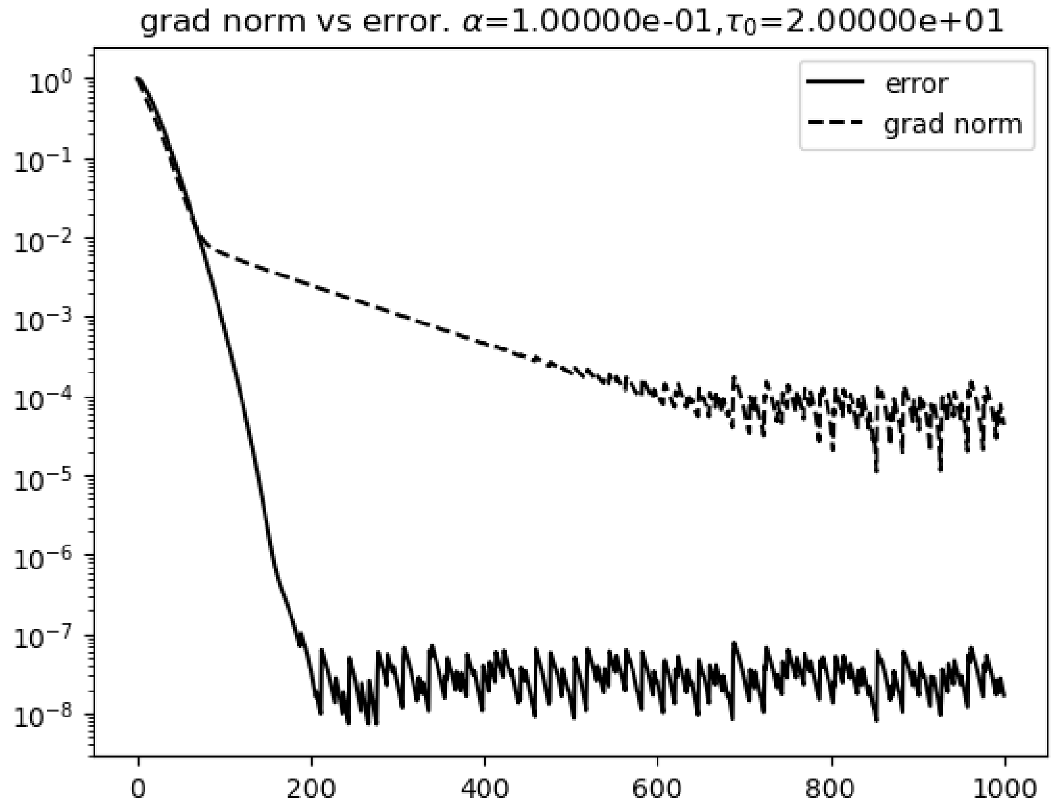


Figures 6ab compare the gradient norm to the error norm for p=.5 and p=.01.

[27]:
```
Figure6ab(15; alpha=.5, tau0=20.0);
```



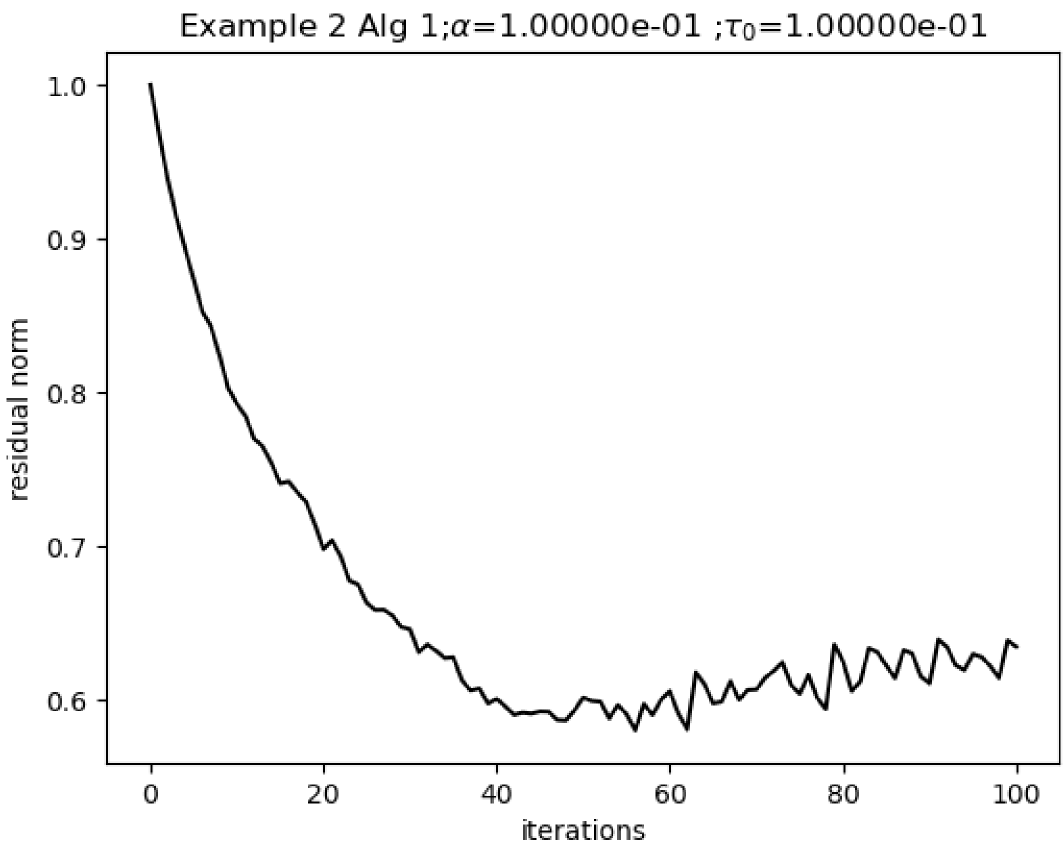
[28]:
```
Figure6ab(15; alpha=.1, tau0=20.0);
```



Here are the two figures for example 2. I am using $\delta = 20$, $\alpha = .1$, and $p = 1.5$, so \$λ> $alpha/\nu$ as we need. This is interesting since it makes it clear that Alg 3 is better.

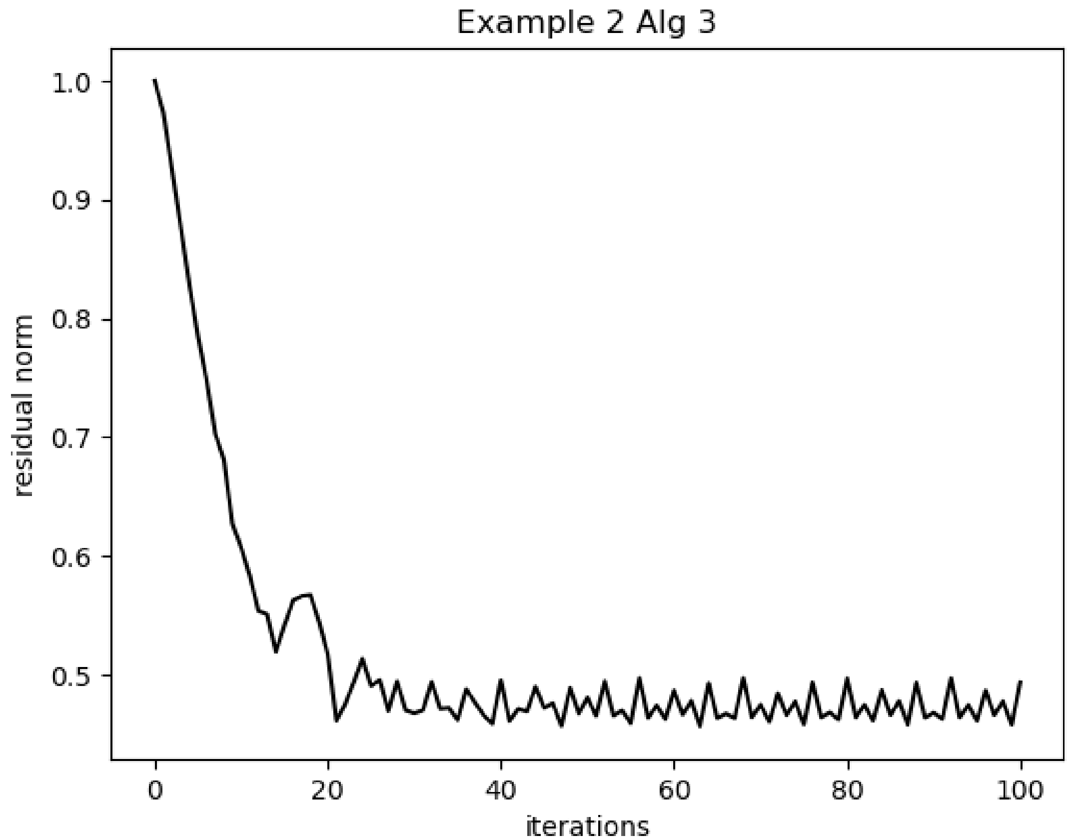
The boundary conditions are $u = u_b$ on the boundary where $u_b(x, y) = .5 - \sin(x)\sin(y)$.

[29]:
```
Example2a(;alpha=.1, tau0=.1);
```



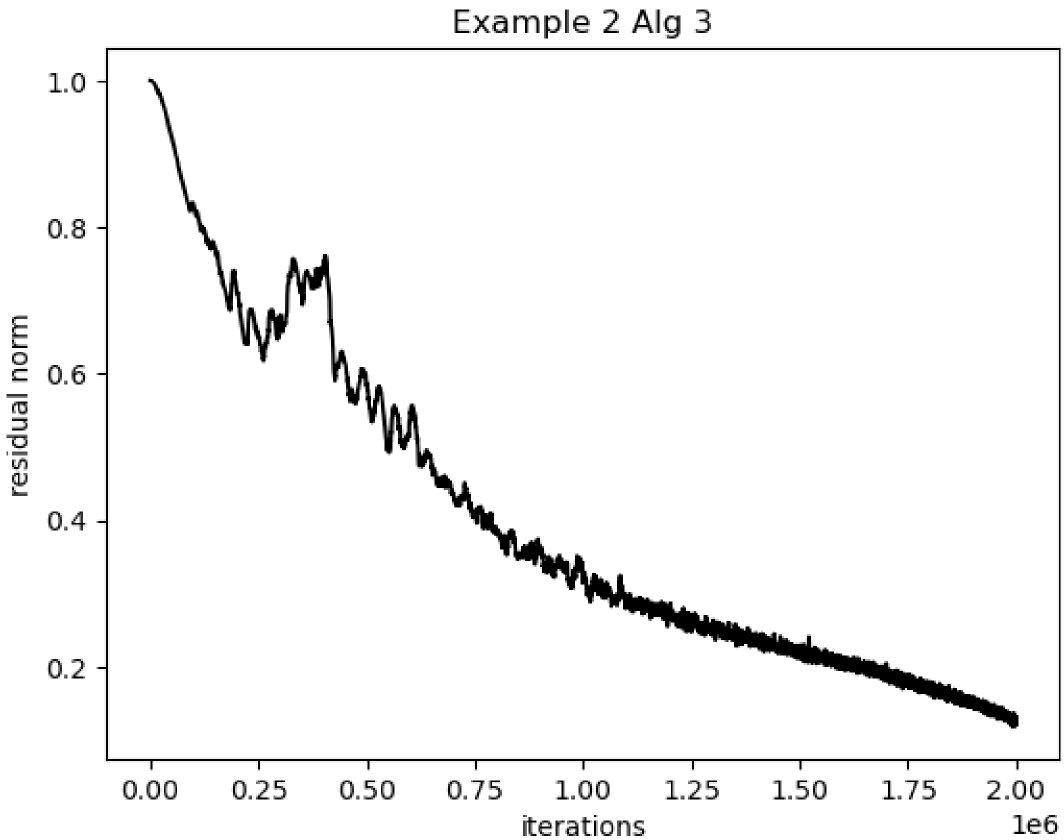
[30]:

```
Example2b(; alpha=.1, tau0=20.0);
```



The next experminet is Alg3 with a larger maxit and a small value of τ_0. The theory predicts that we will get a smaller gradient norm.

[31]:
```
Example2b(;alpha=.1, tau0=.001, maxit=2000000)
```



[31]: Python: Text(0.5, 1.0, 'Example 2 Alg 3')

[ ]: