Examples

January 19, 2026

[18]:
```
include("notebook_init.jl");
```

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

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?

1 Example 1

This problem is to solve the following two-dimensional PDE,

$$\mathcal{F}(u) = -\Delta u + \gamma u_+^\alpha = 0,$$

where $\alpha \in (0,1)$, $\gamma > 0$ is a constant and $u_+ = \max\{u, 0\}$. Discretizing this problem with the standard five point scheme leads to

$$\mathbf{F}(\mathbf{u}) = \mathbf{A}\mathbf{u} + \gamma \mathbf{u}_+^\alpha - \mathbf{b} = 0$$

$\mathbf{A} \in \mathbb{R}^{n\times n}$ is the discretization of $-\Delta$ with zero boundary conditions, $\mathbf{b} \in \mathbb{R}^n$ encodes the boundary conditions, and $\mathbf{u}_+^\alpha = \max\{\mathbf{u}, 0\}^\alpha$ is understood as a component-wise operation.

We now modify the above problem to enable direct computation of errors in the iterations. To this end, we take as the exact solution the function

$$u^*(x,y) = \left(\frac{3r-1}{2}\right)^2 \max\left\{0, r - \frac{1}{3}\right\}$$

where $r = \sqrt{x^2+y^2}$, and enforce the boundary conditions

$$u(x,1) = u^*(x,1),\ u(x,0) = u^*(x,0),\ u(1,y) = u^*(1,y),\ u(0,y) = u^*(0,y),$$

for $0 < x, y < 1$. Hence our modified equation is

$$\mathbf{F}(\mathbf{u}) - \mathbf{c}^* = 0,$$

where $\mathbf{c}^* = \mathbf{F}(\mathbf{u}^*)$.

The nonlinear system is first order optimality condition for the strongly convex optimization problem.

$$\min_{\mathbf{u}\in\mathbb{R}^n} \ f(\mathbf{u}) = \frac{1}{2}\mathbf{u}^\top \mathbf{A}\mathbf{u} + \frac{\gamma}{1+\alpha}\mathbf{e}^\top \mathbf{u}_+^{1+\alpha} - (\mathbf{b}+\mathbf{c}^*)^\top \mathbf{u},$$

where $\mathbf{e} \in \mathbb{R}^n$ is the vector of all ones.

1.1 Figures for 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.

Figure 1a

[19]:
```
Figure1_2a(15; alpha=.5);
```

```
Alg1: testing $\tau_0$. n=15,$\alpha$ =5.00000e-01
```

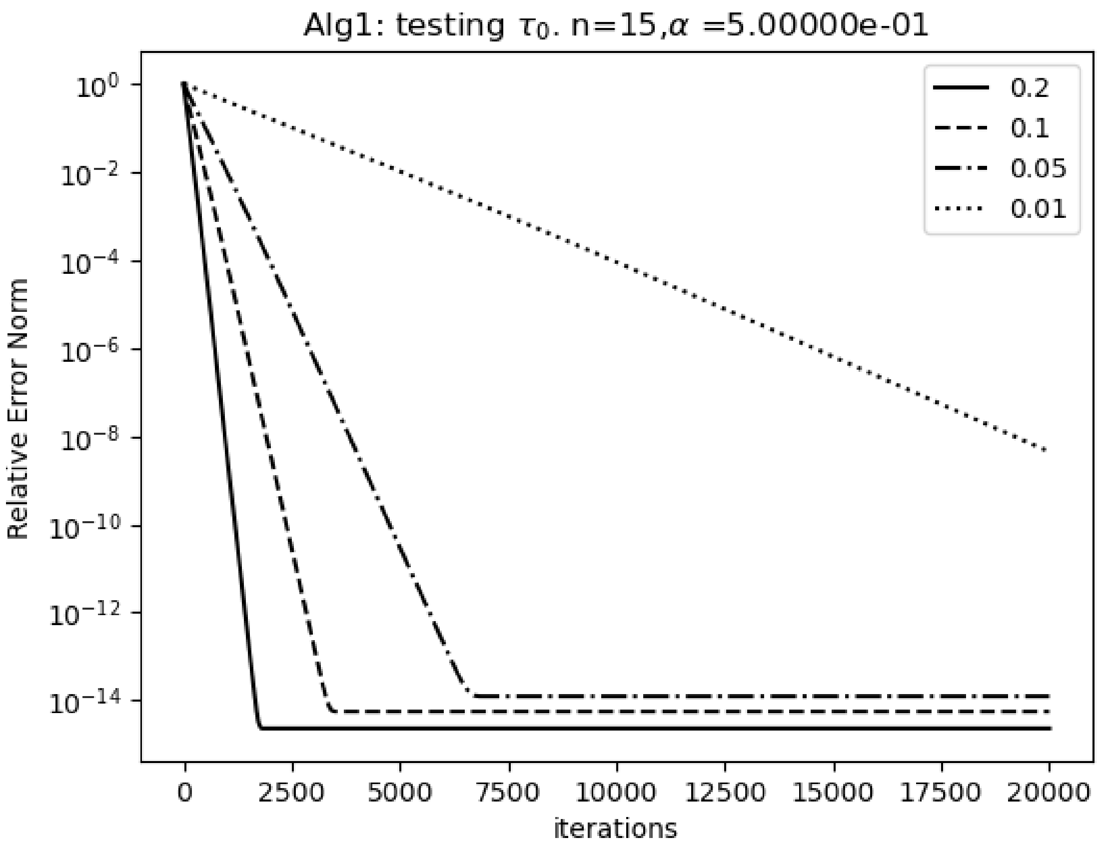
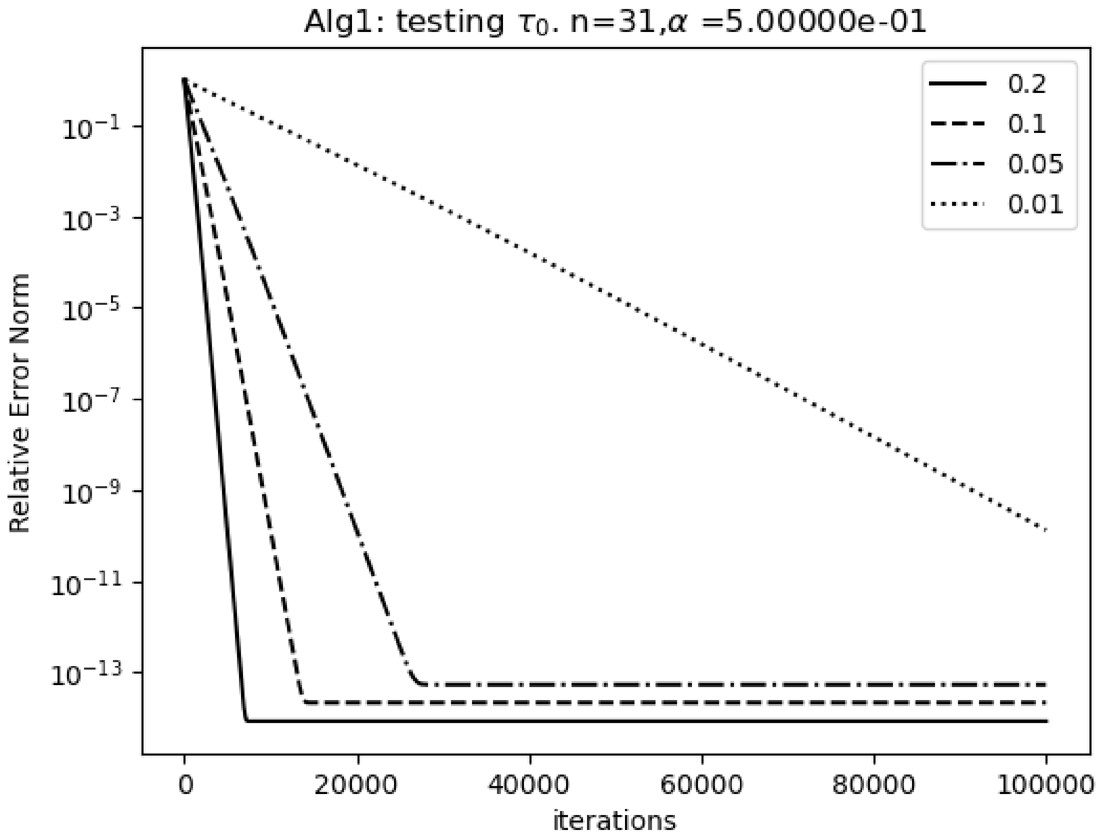


Figure 2a

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

[20]:
```
Figure1_2a(31);
```

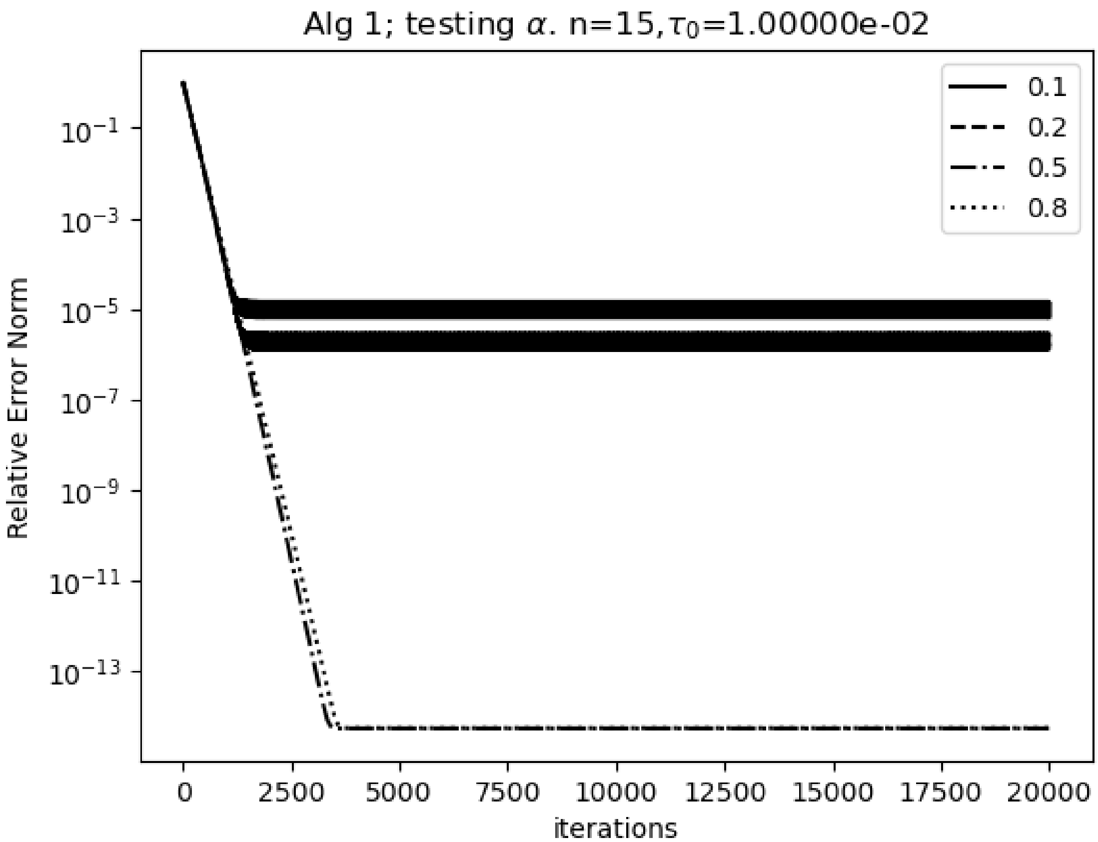
```
Alg1: testing $\tau_0$. n=31,$\alpha$ =5.00000e-01
```



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.

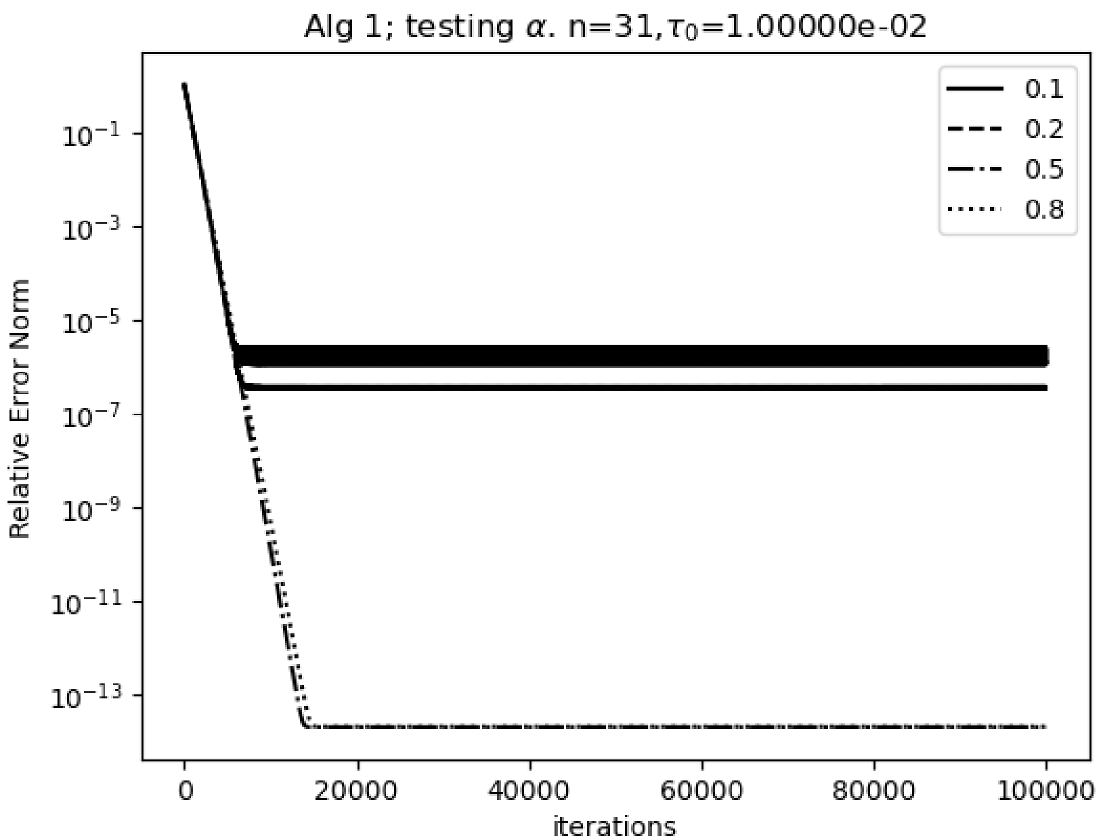
Figure 1b

[21]:
```
Figure1_2b(15);
```



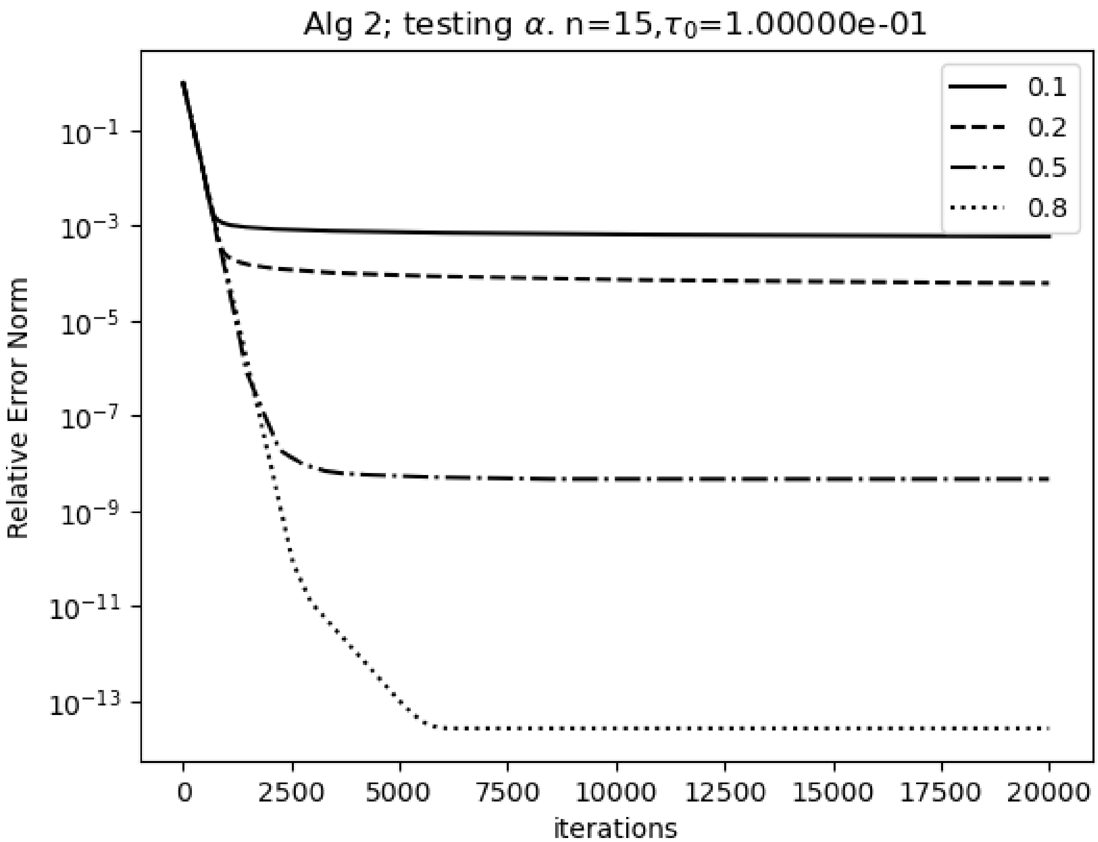
And finally repeat the computation for a 31x31 grid. This will complete the computations for Example 1 + Algorithm 1.

[22]:
```
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.

[23]:
```
Figure3(15);
```

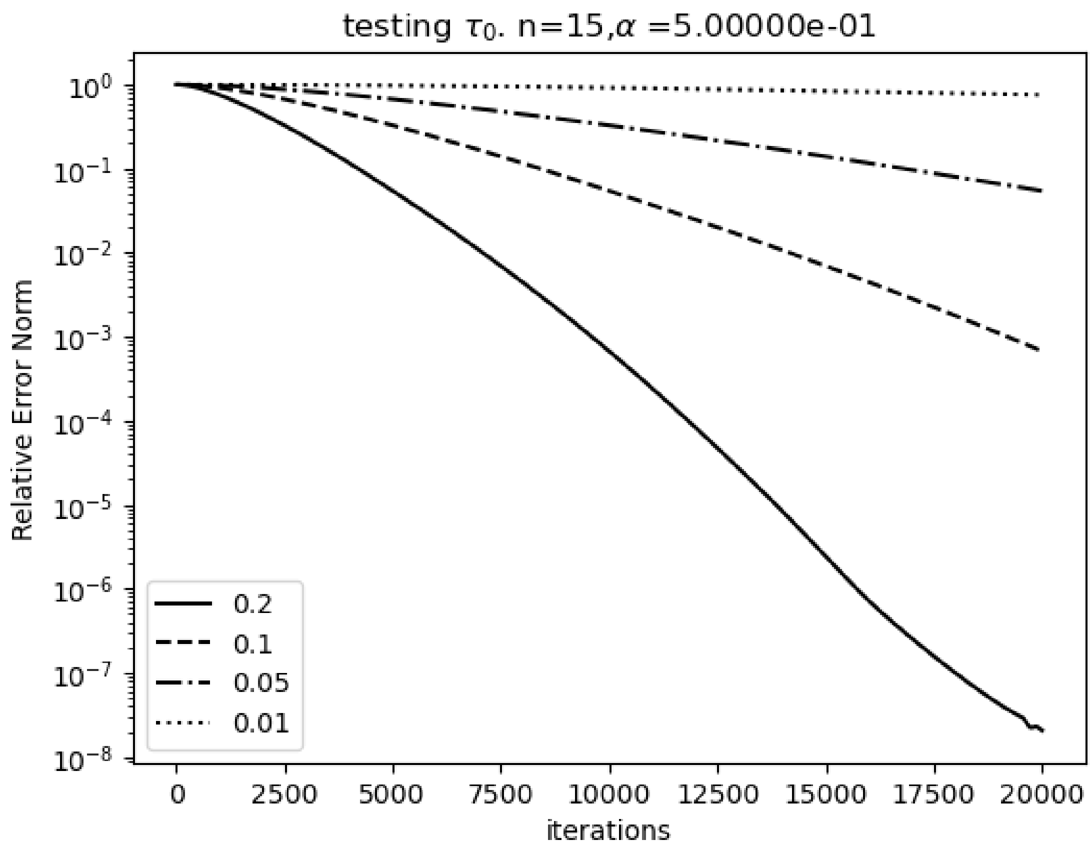


The advantage of the line search is that one does not have to manually adjust τ_0.

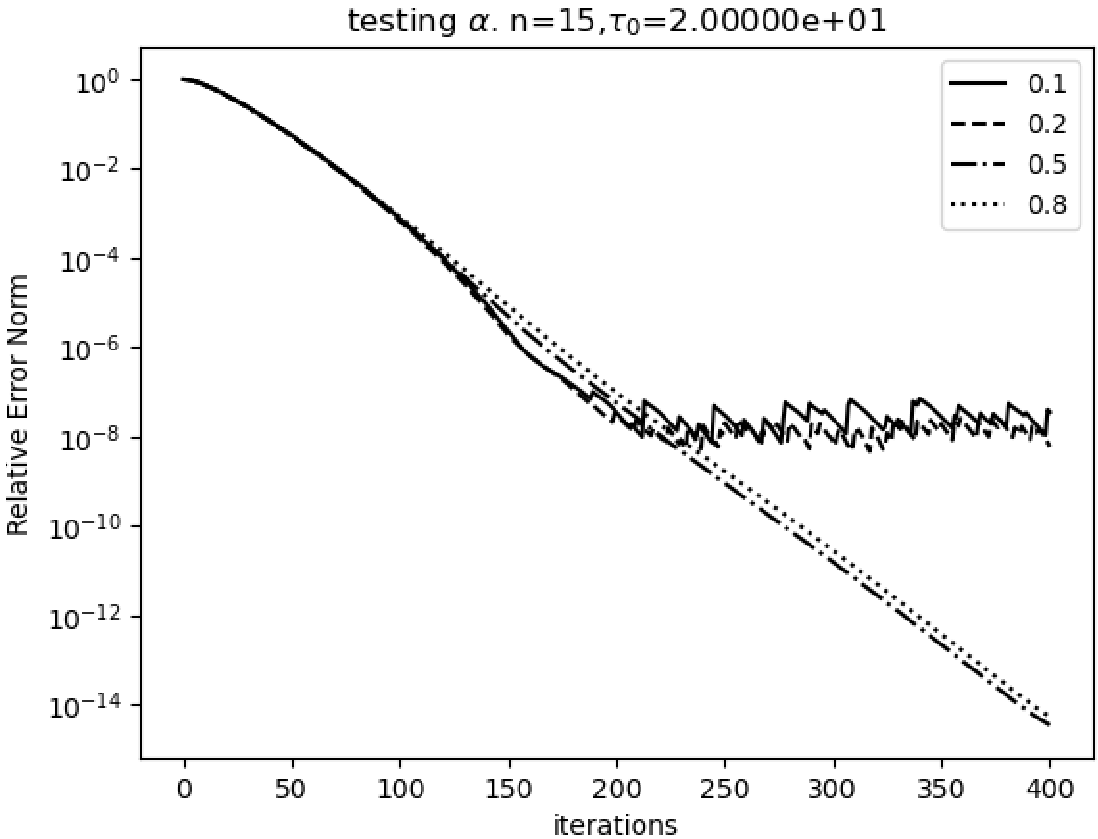
The results for Algoirthm 3 are in Figures 4 and 5. We set $\nu = \tau_0 h^2$ in these examples and will need to modify that to use the estimate in Remark 4.2. In the first two figures 4(a) and 4(b) use use the values of τ_0 we use in Figure 1.

[24]:
```
Figure4_5a(15);
```

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



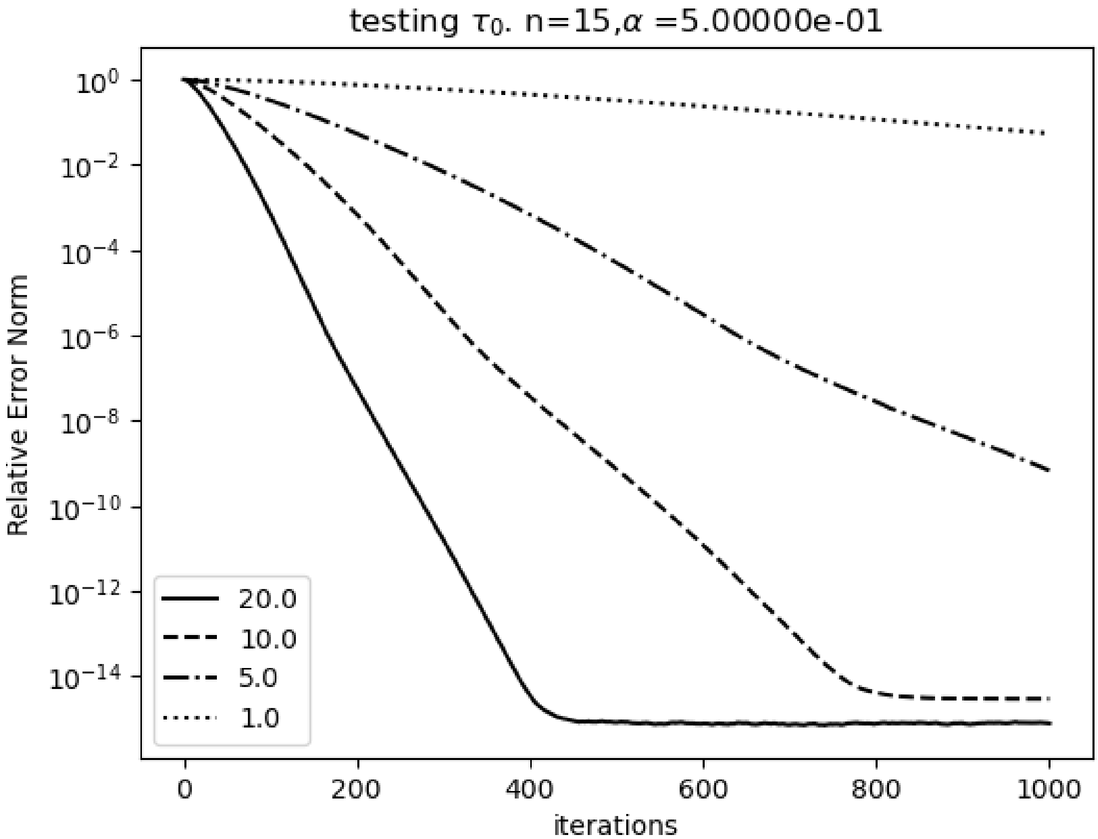
[25]:
```
Figure4_5b(15);
```



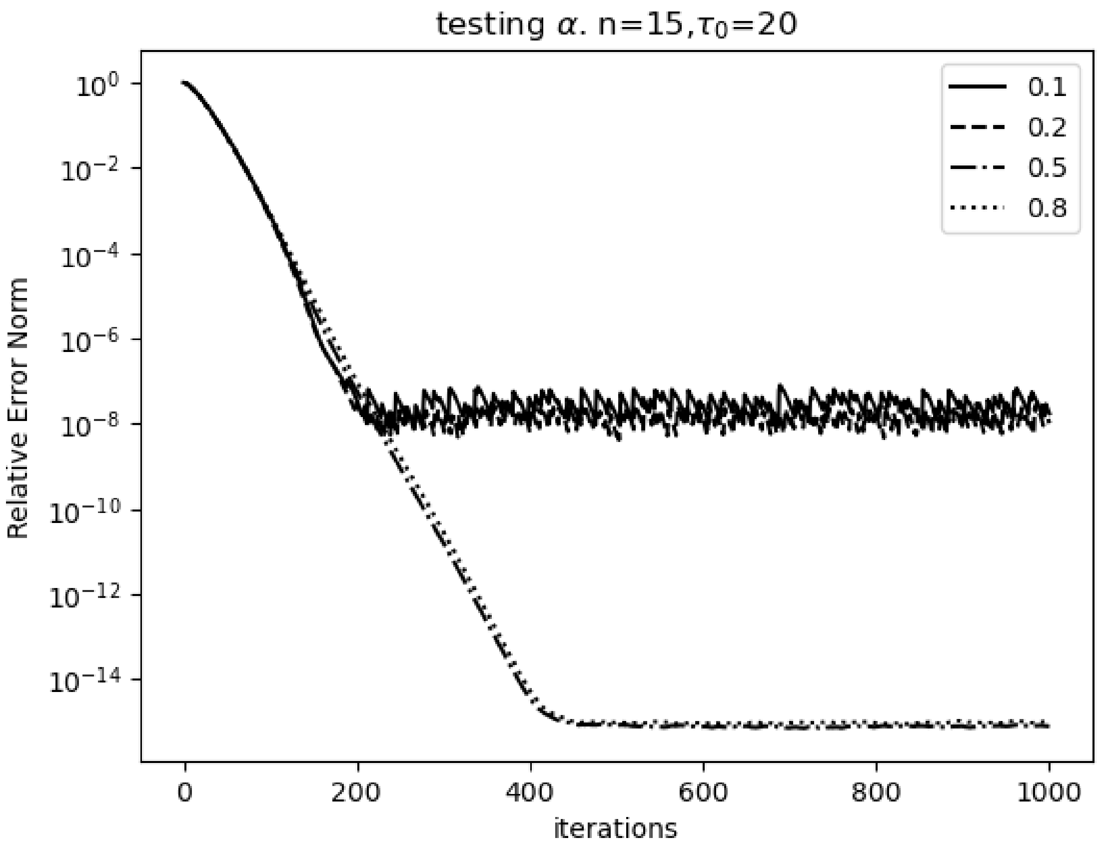
Now we use the larger values of τ_0.

[26]:
```
Figure4_5a(15; maxit=1000, tauvec=[20.0, 10.0, 5.0, 1.0]);
```

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

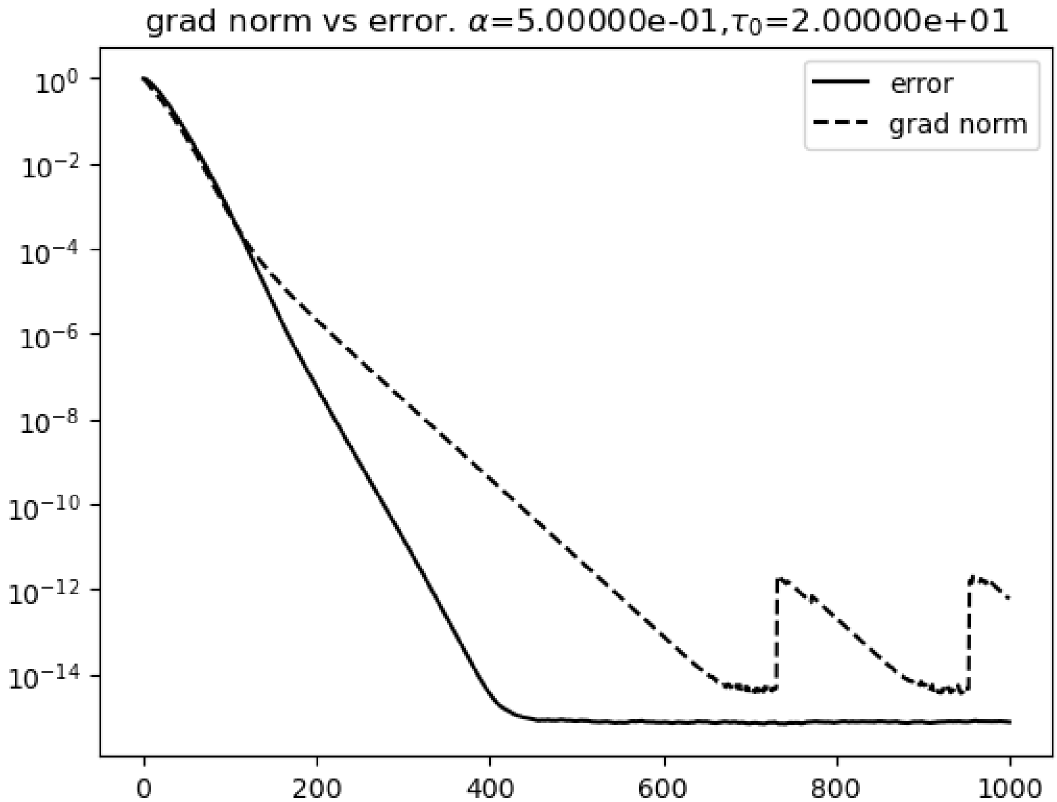


[27]:
```
Figure4_5b(15; maxit=1000, tau0=20);
```

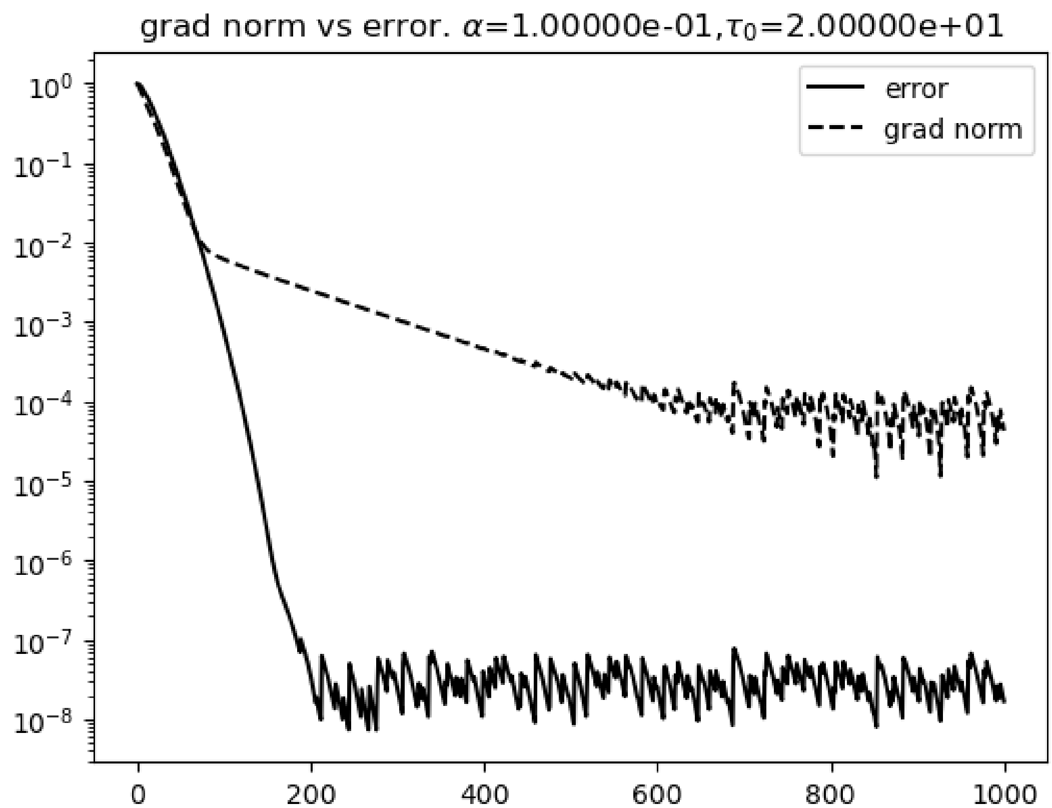


Figures 6ab compare the gradient norm to the error norm for p=.5 and p=.01.

[28]:
```
Figure6ab(15; alpha=.5, tau0=20.0);
```



[29]:
```
Figure6ab(15; alpha=.1, tau0=20.0);
```



2 Example 2

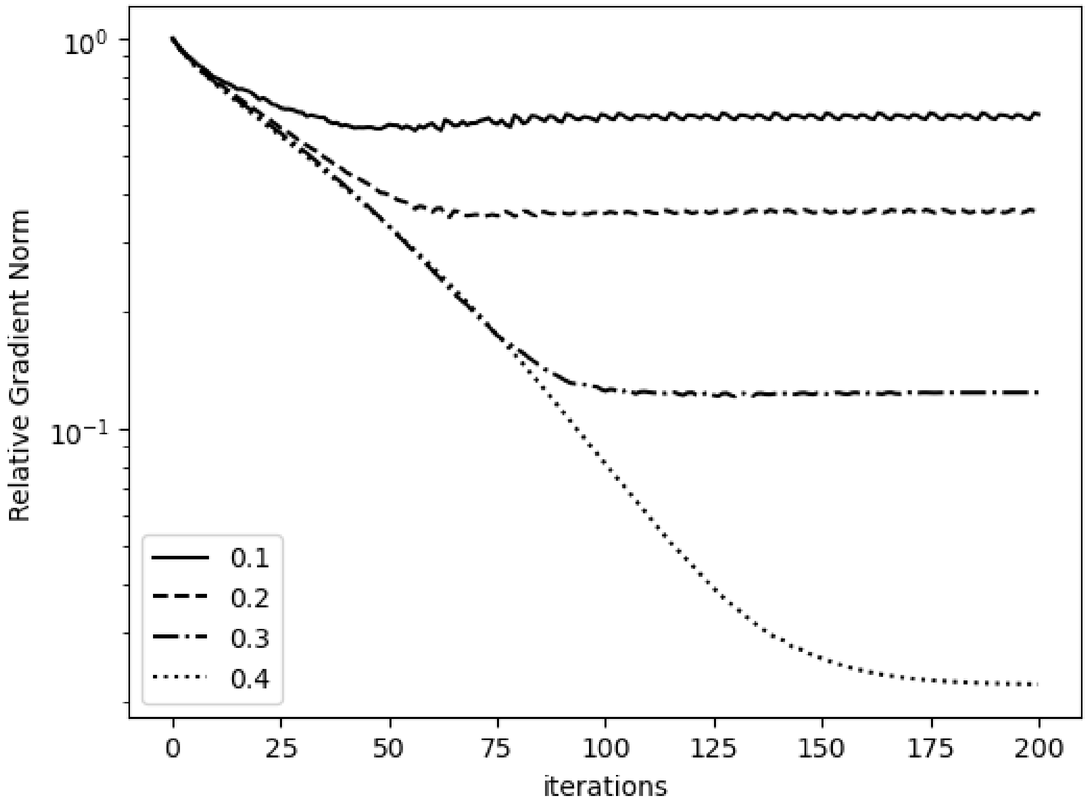
The boundary conditions are $u = u_b$ on the boundary where $u_b(x, y) = .5 - \sin(x)\sin(y)$. This choice insures that the solution will change signs in the domain so the non-Lipschitz features are exercised.

2.1 Figures for Example 2

In all the examples we use $\delta = 20$ and $p = 1.5$. The parameter α ranges from .1 to .8, so $\delta > \alpha/p$ in all cases.

Figure7a

[30]:
```
Figure78a(15; tau0=.1, maxit=200, pvec=[.1, .2, .3, .4])
```



[30]: Python: Text(0.5, 24.0, 'iterations')

Figure7b.

[31]:
```
Figure78b(15; tau0=20.0, pvec=[.1, .2, .3, .4], maxit=100);
```

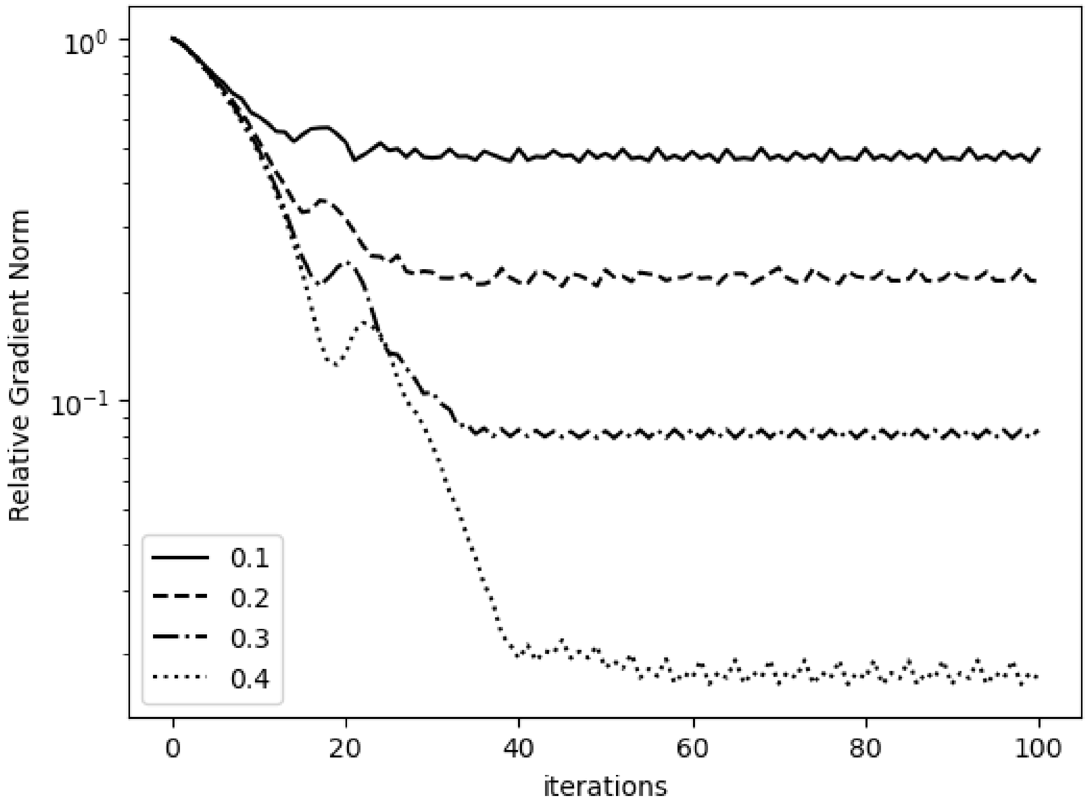


Figure 8a

[32]:
```
Figure78a(15; tau0=.1, pvec=[.5, .6, .7, .8], maxit=2000);
```

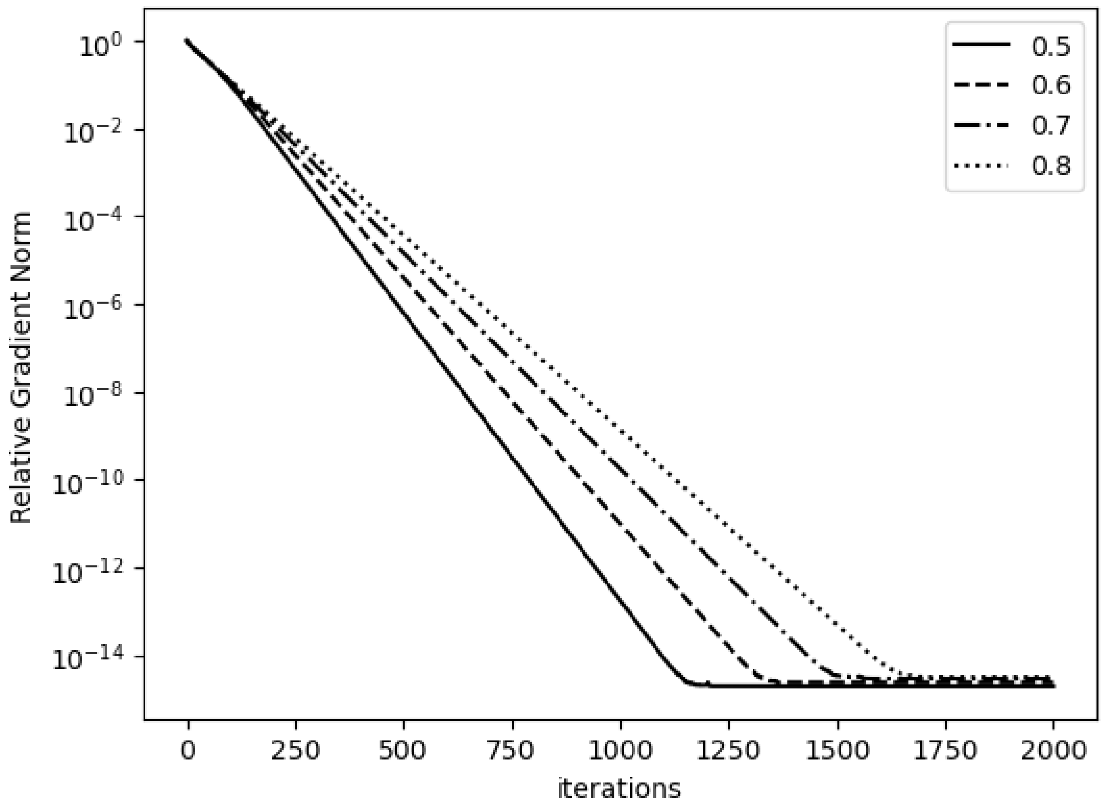
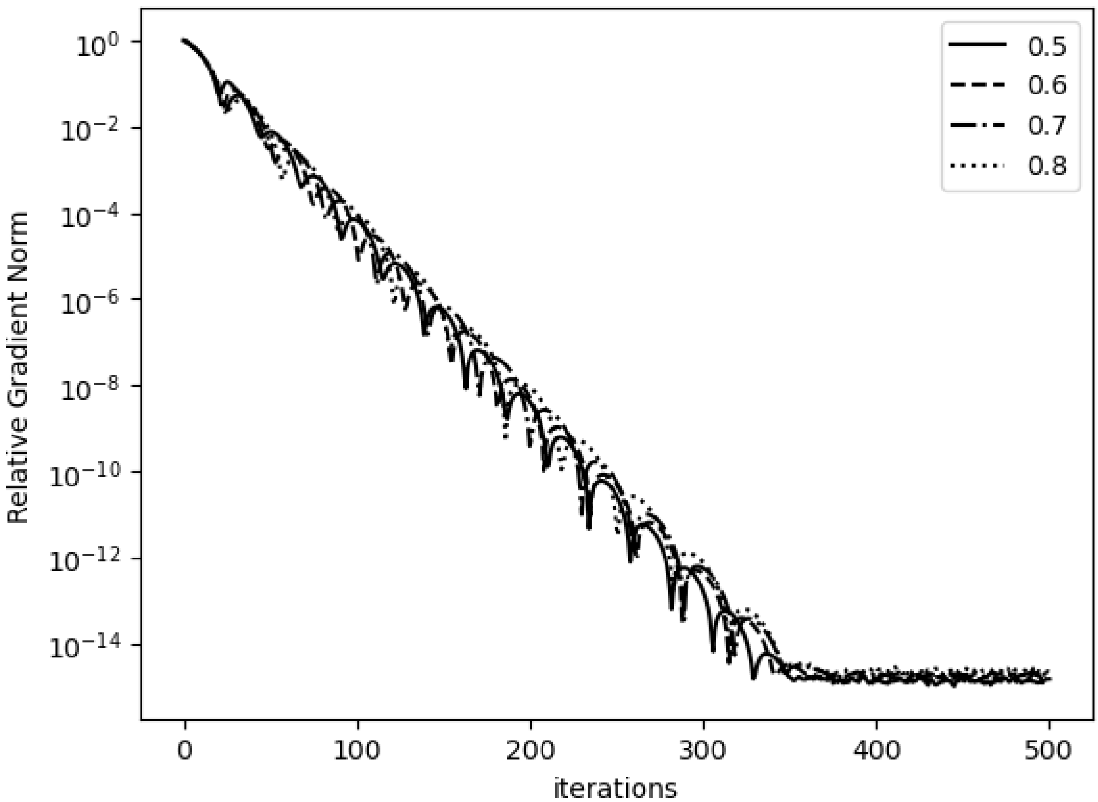


Figure 8b.

```
[33]: Figure78b(15; tau0=20.0, pvec=[.5, .6, .7, .8], maxit=500);
```



Relative Gradient Norm
iterations
0.5
0.6
0.7
0.8

[ ]: