Examples

January 21, 2026

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

This notebook generates the figures in the paper *Complexity of Projected Gradient Methods for Strongly Convex Optimization with Hölder Continuous Gradient Terms*

by X. Chen, C. T. Kelley, and L. Wang

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.

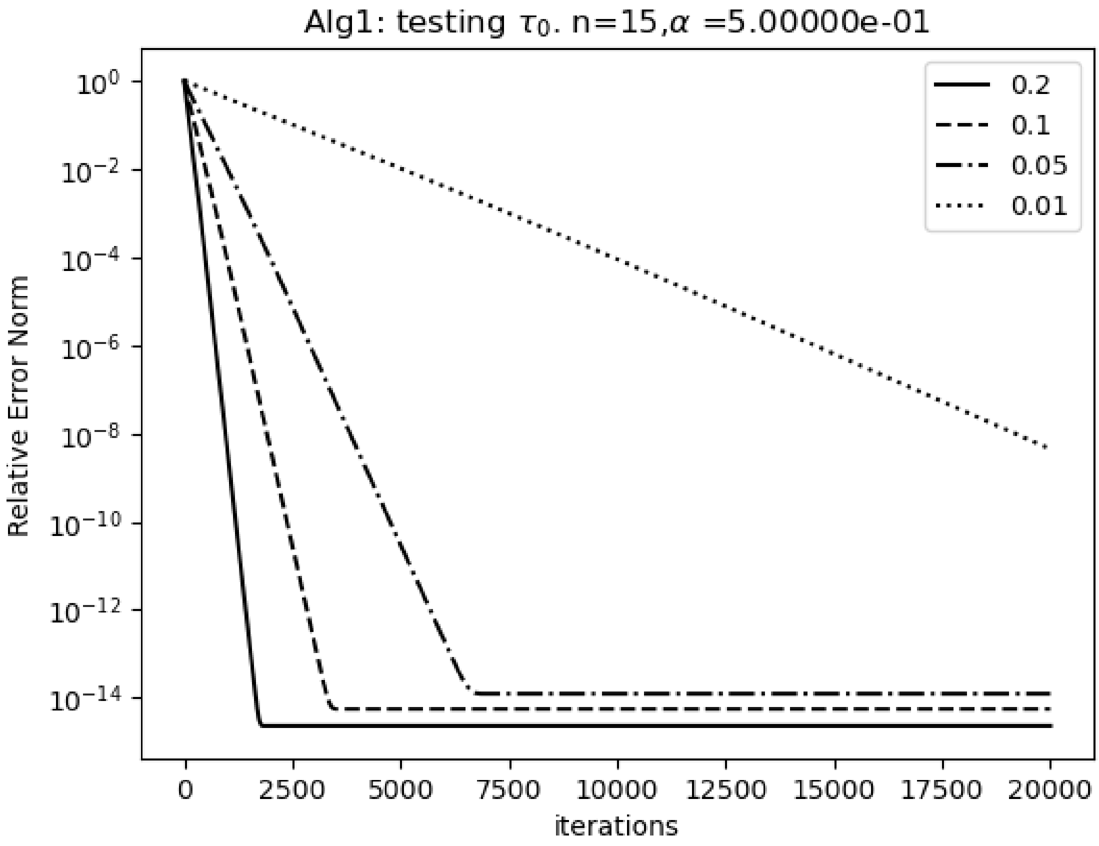
The files for building the figures are in /**src**/**Figures**. The code is **Figures_Alg1.jl** and the functions **Figure1_2a** and **Figure1_2b**. The functions take the dimension as an argument.

Figure 1a

[31]:

```
Figure1a(15; alpha=.5);
```

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

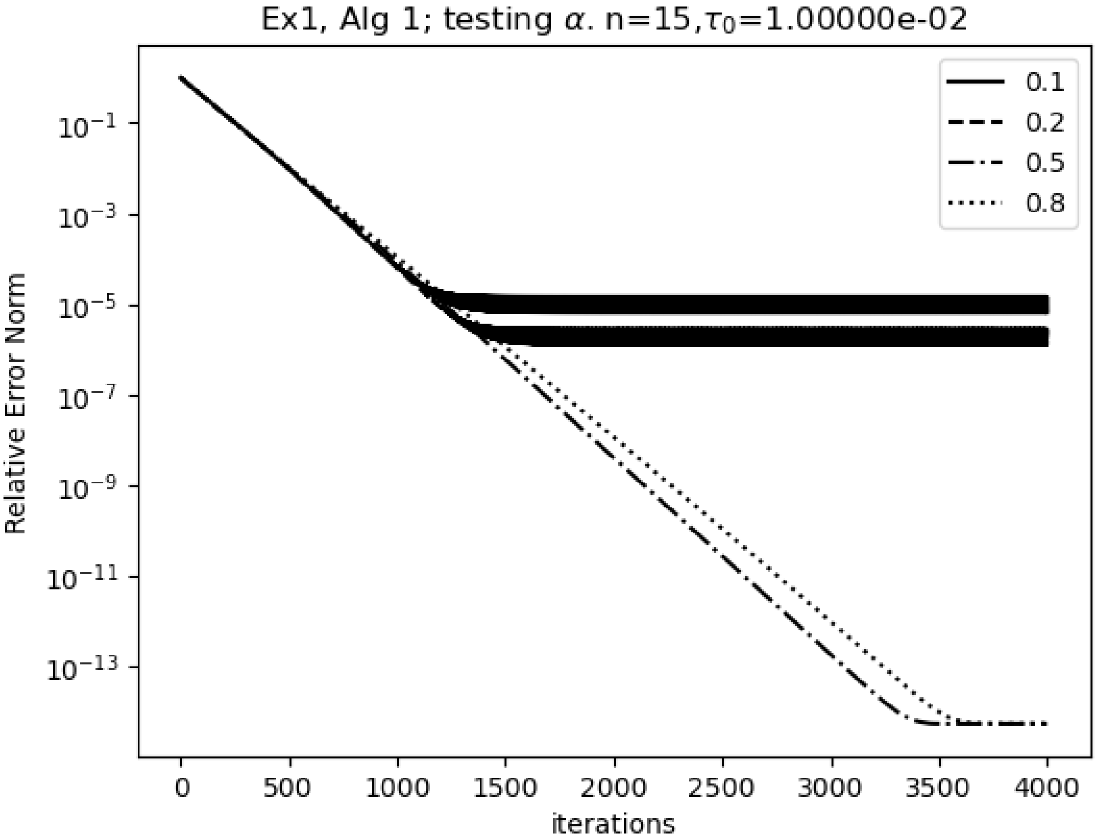


Now we compare the effects of changing the exponent α. Here we can see the effects of the change I made in Alg 1 by letting the gradient norm increase without terminating the iteration. **In this figure, and in several others, the thick lines are artifacts of small rapid oscillations in the error norm which reflect stagnation in the iteration.**

Figure 1b

[32]:

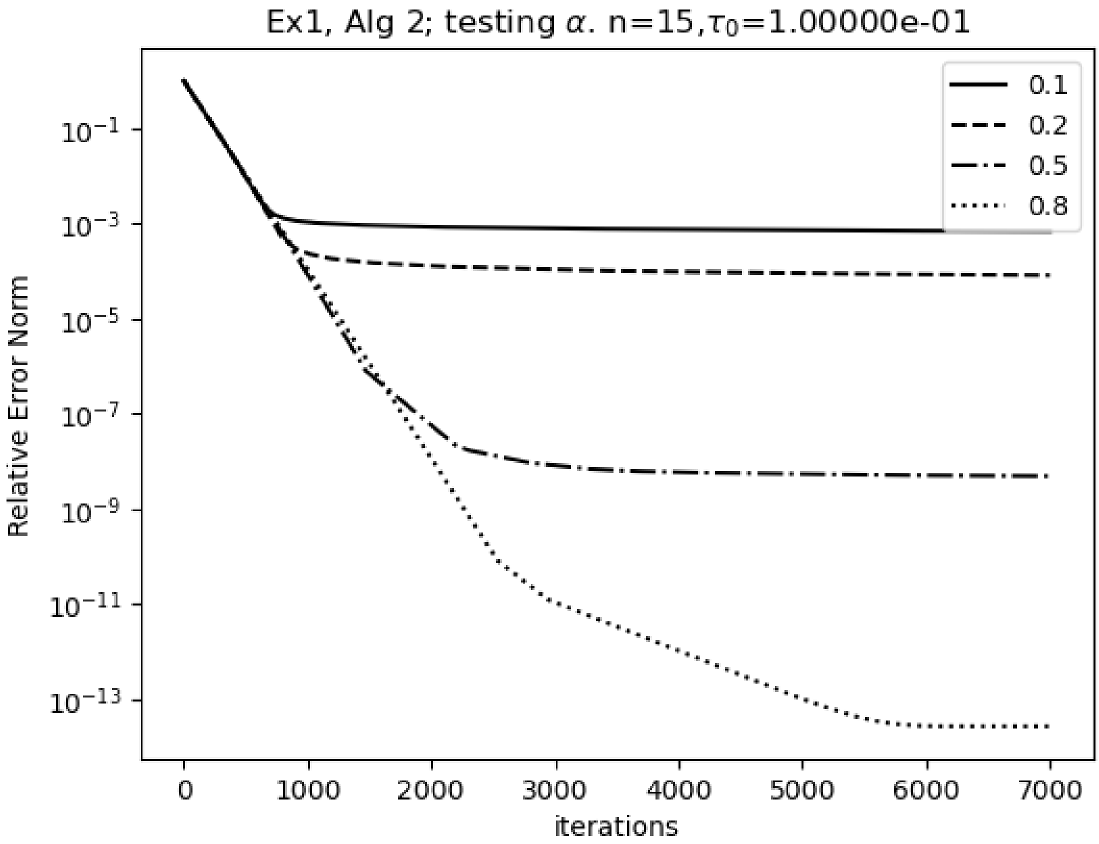
```
Figure1b(15);
```



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

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

[33]:
```
Figure2(15; maxit=7000);
```

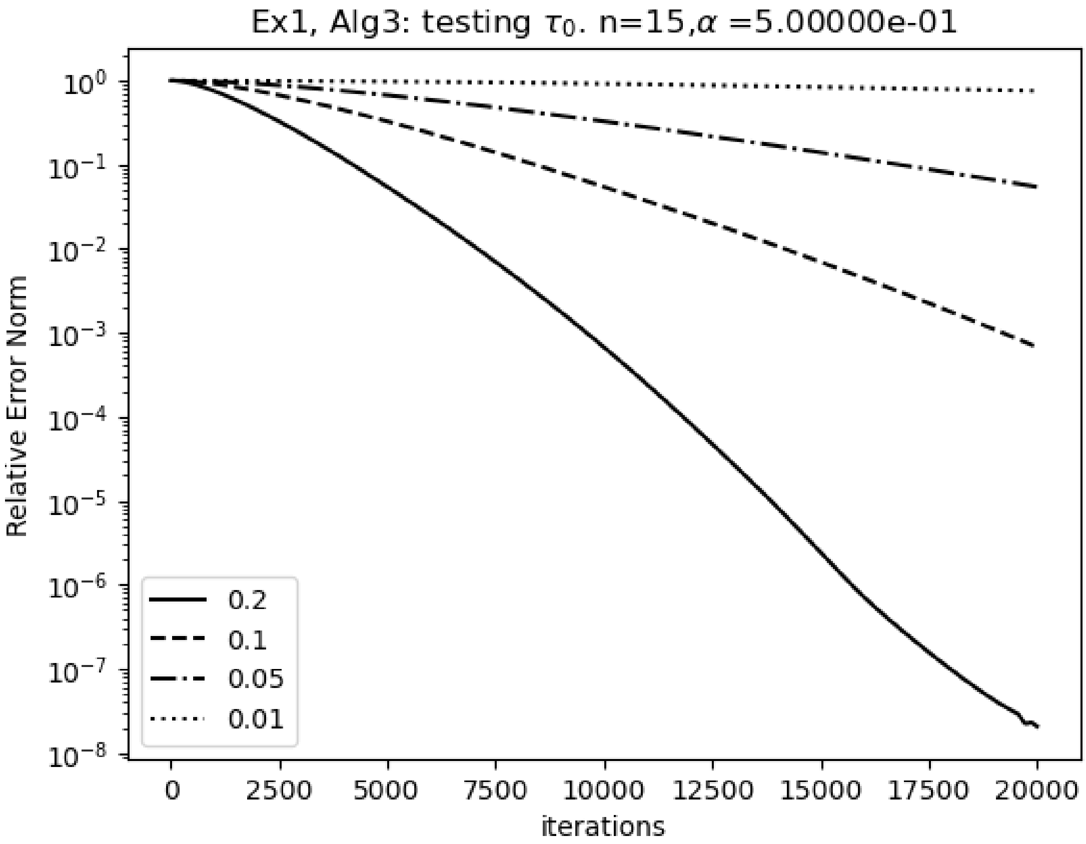


The advantage of the line search is that one does not have to manually adjust τ_0.

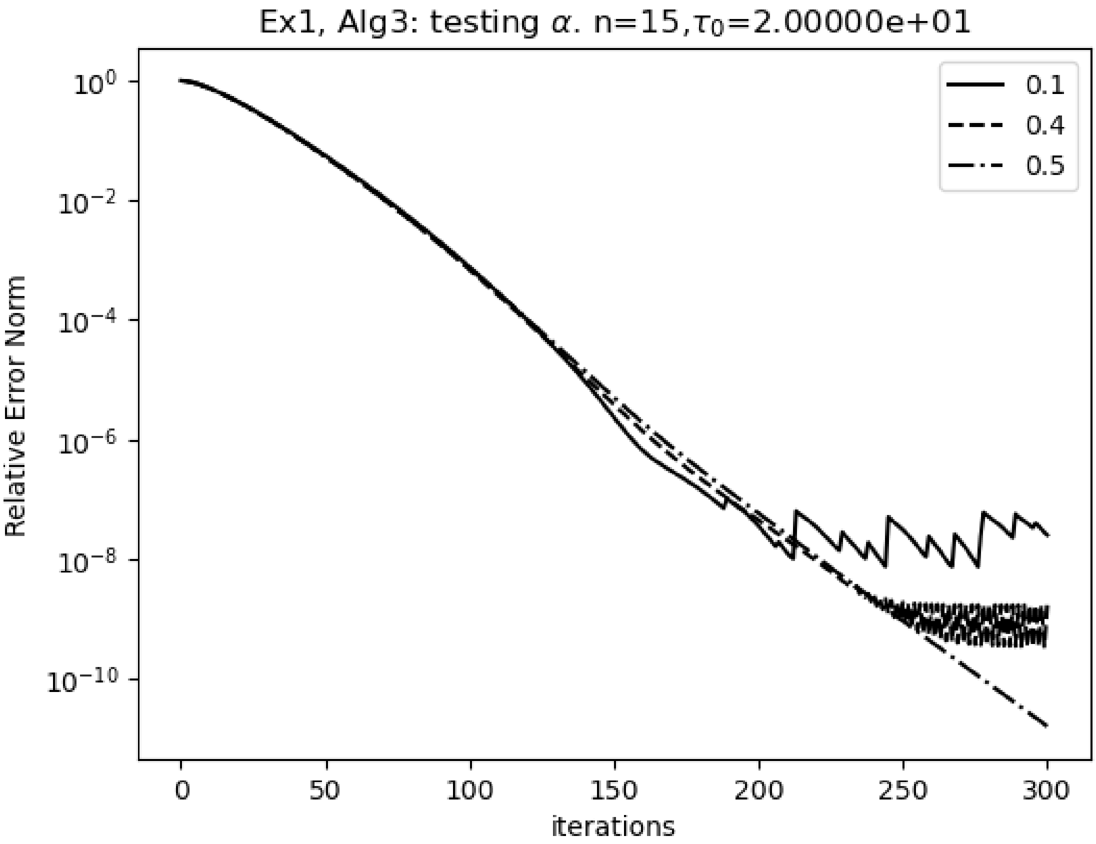
The results for Algoirthm 3 are in Figures 4 and 5. We set $\nu = \tau_0 h^2$ in these examples and will need to modify that to use the estimate in Remark 4.2. In the first two figures 3(a) and 3(b) use use the values of τ_0 we used in Figure 1.

```
[34]: Figure3_4a(15);
```

```
Ex1, Alg3: testing $\tau_0$. n=15,$\alpha$ =5.00000e-01
```



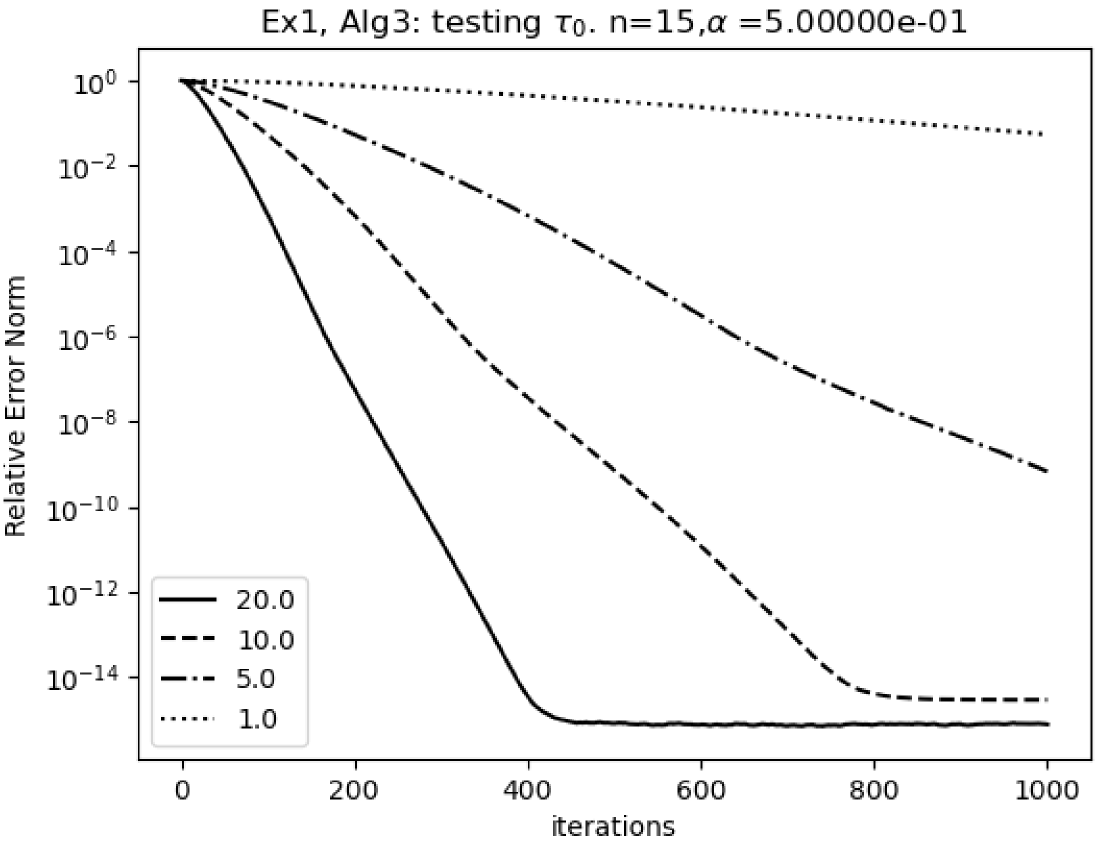
[35]:
```
Figure3_4b(15; maxit=300);
```



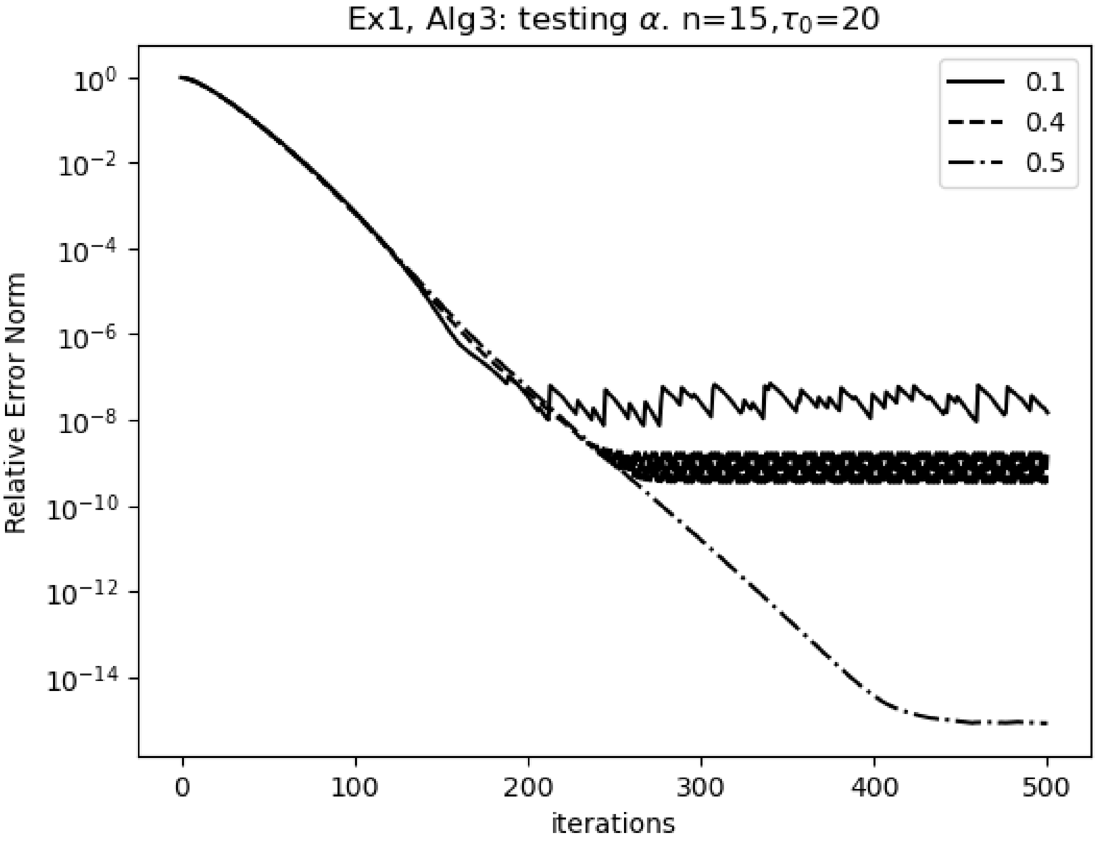
Now we use the larger values of τ_0.

```
[36]: Figure3_4a(15; maxit=1000, tauvec=[20.0, 10.0, 5.0, 1.0]);
```

```
Ex1, Alg3: testing $\tau_0$. n=15,$\alpha$ =5.00000e-01
```



```
[37]: Figure3_4b(15; maxit=500, tau0=20);
```



2 Example 2

This example is a constrained semi-linear elliptic boundary value problem. Define for sufficienlty smooth u

$$\mathcal{H}(u) = -\Delta u + \delta |u|^{\alpha} \mathsf{sign}(u) - |u|^{p-1} u$$

on $D = (0,1)^2$ with the boundary condition $u(x,y) = 0.5 - \sin(x)\sin(y)$ on ∂D. Here, $\alpha \in (0,1)$, $p > 1$, and $\delta > p/\alpha$ are three constants. We consider the variational inequality that is to find u^* with $u * (x,y) \in [-1,1]$ such that

$$\mathcal{H}(u^*)(u - u^*) \geq 0$$

for any u with values in $[-1,1]$.

This problem is equivalent to the following nonlinear equation,

$$0 = \mathcal{F}(u) := \begin{cases} \mathcal{H}(u), & \text{if } u - \mathcal{H}(u) \in [-1,1], \\ u - 1, & \text{if } u - \mathcal{H}(u) \geq 1, \\ u + 1, & \text{otherwise.} \end{cases}$$

After discretization we have the nonlinear system

$$0 = \mathbf{F}(\mathbf{u}) := \Big(\mathbf{u} - \Pi_{\mathbf{U}} \Big(\mathbf{u} - \tau \Big(\mathbf{A}\mathbf{u} + \delta \left|\mathbf{u}\right|^{\alpha} \mathsf{sign}(\mathbf{u}) - \left|\mathbf{u}\right|^{p-1} \mathbf{u} - \mathbf{b} \Big) \Big) \Big)$$

where $\mathbf{U} = [-1, 1]^n$, $\tau > 0$ is the stepsize, $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a symmetric positive definite matrix, and $\mathbf{b} \in \mathbb{R}^n$ encodes the boundary conditions. The boundary conditions insure that the solution will change signs in the domain so the non-Lipschitz features are exercised. Hence the iteration for Algorithm 1 would look like

$$\mathbf{u}_{k+1} = \Pi_{\mathbf{U}} \left(\mathbf{u} - \tau \left(\mathbf{A}\mathbf{u}_k + \delta \left| \mathbf{u}_k \right|^{\alpha} \mathsf{sign}(\mathbf{u}_k) - \left| \mathbf{u}_k \right|^{p-1} \mathbf{u}_k - \mathbf{b} \right) \right)$$

In the example we use $\tau = \tau_0 h^{-2}$ as we did in Example 1.

The nonlinear equation is the necessary condition for the optimization problem

$$\min_{\mathbf{u} \in \Omega} f(\mathbf{u}) := \frac{1}{2} \left(f_1(\mathbf{u}) + f_2(\mathbf{u}) \right)$$

where

$$f_1(\mathbf{u}) = \mathbf{u}^\top \mathbf{A}\mathbf{u} - 2\mathbf{b}^\top \mathbf{u} - \frac{2}{1+p} \mathbf{e}^\top \left| \mathbf{u} \right|^{1+p}, \text{ and } f_2(\mathbf{u}) = \frac{2\delta}{1+\alpha} \mathbf{e}^\top \left| \mathbf{u} \right|^{1+\alpha}.$$

Our conditions on the problem struture hold with $\alpha_1 = 1$, $L_1 = 2 \left\| \mathbf{A} \right\| + 2p$, $\alpha_2 = \alpha$, and $L_2 = 2\delta\alpha$. The condition that $\alpha \in (0, 1)$, $p > 1$, and $\delta > p/\alpha$ implies strong convexity.

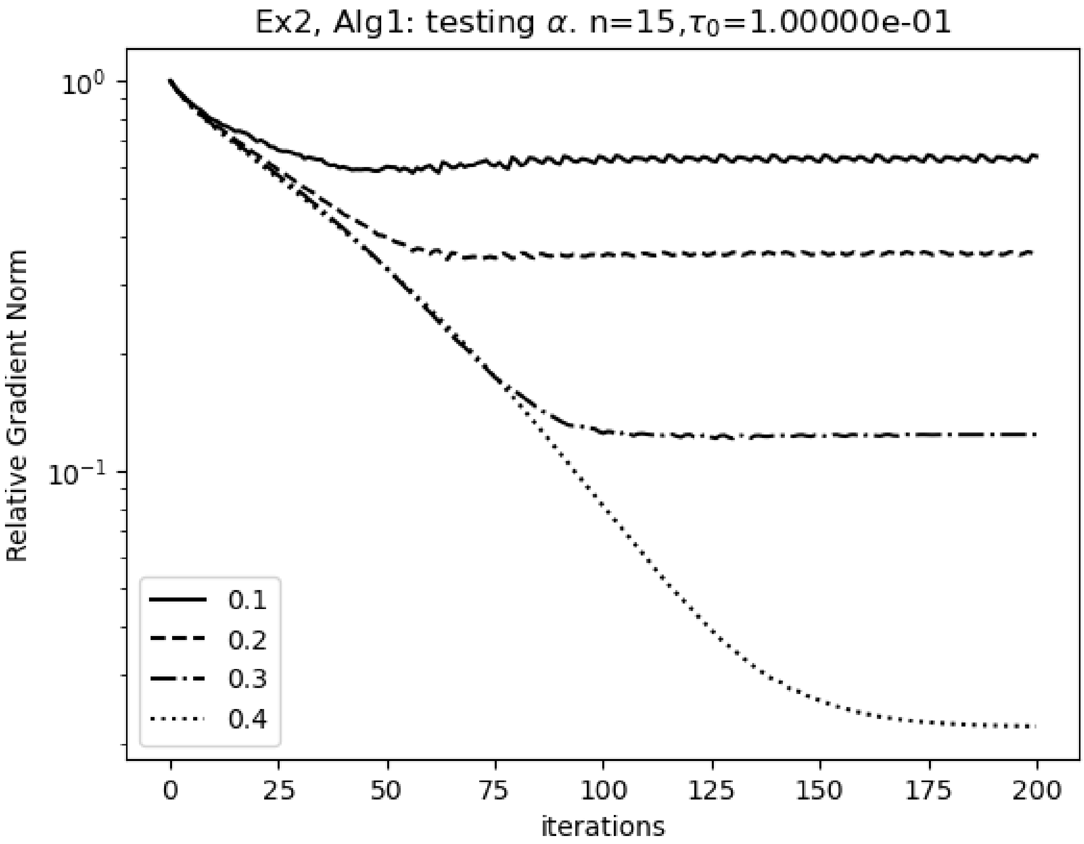
2.1 Figures for Example 2

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.

Figure5a

[38]:

```
Figure5ac(15; tau0=.1, maxit=200, pvec=[.1, .2, .3, .4])
```



```
[38]: Python: Text(0.5, 1.0, 'Ex2, Alg1: testing $\\alpha$.
      n=15,$\\tau_0$=1.00000e-01')
```

Figure5b.

```
[39]: Figure5bd(15; tau0=20.0, pvec=[.1, .2, .3, .4], maxit=50);
```

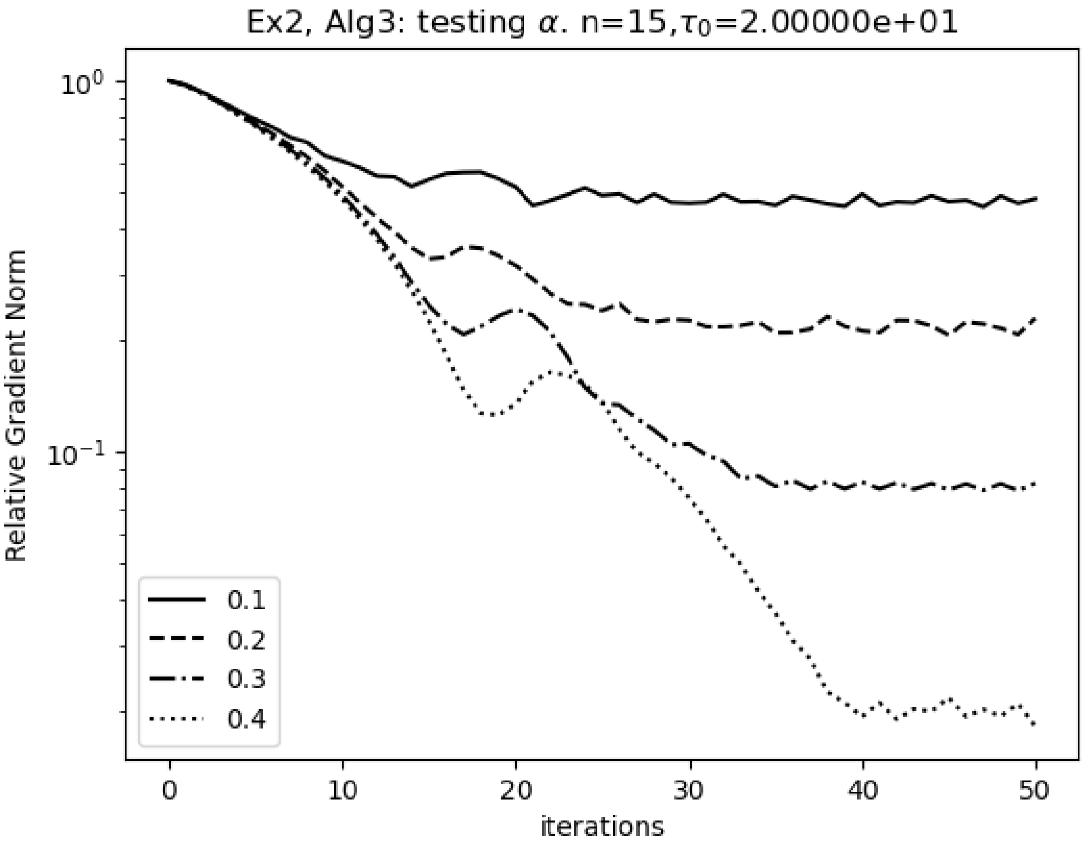


Figure 5c.

```
[40]: Figure5ac(15; tau0=.1, pvec=[.5, .6, .7, .8], maxit=2000);
```

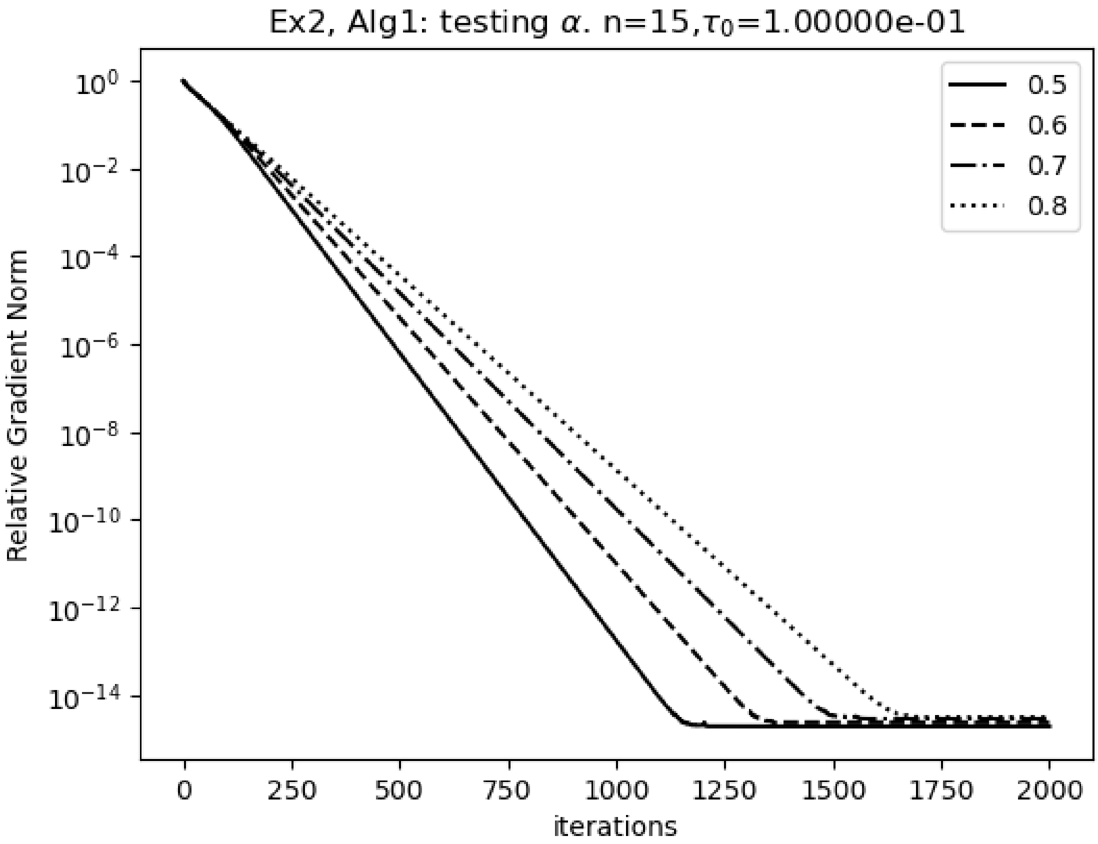
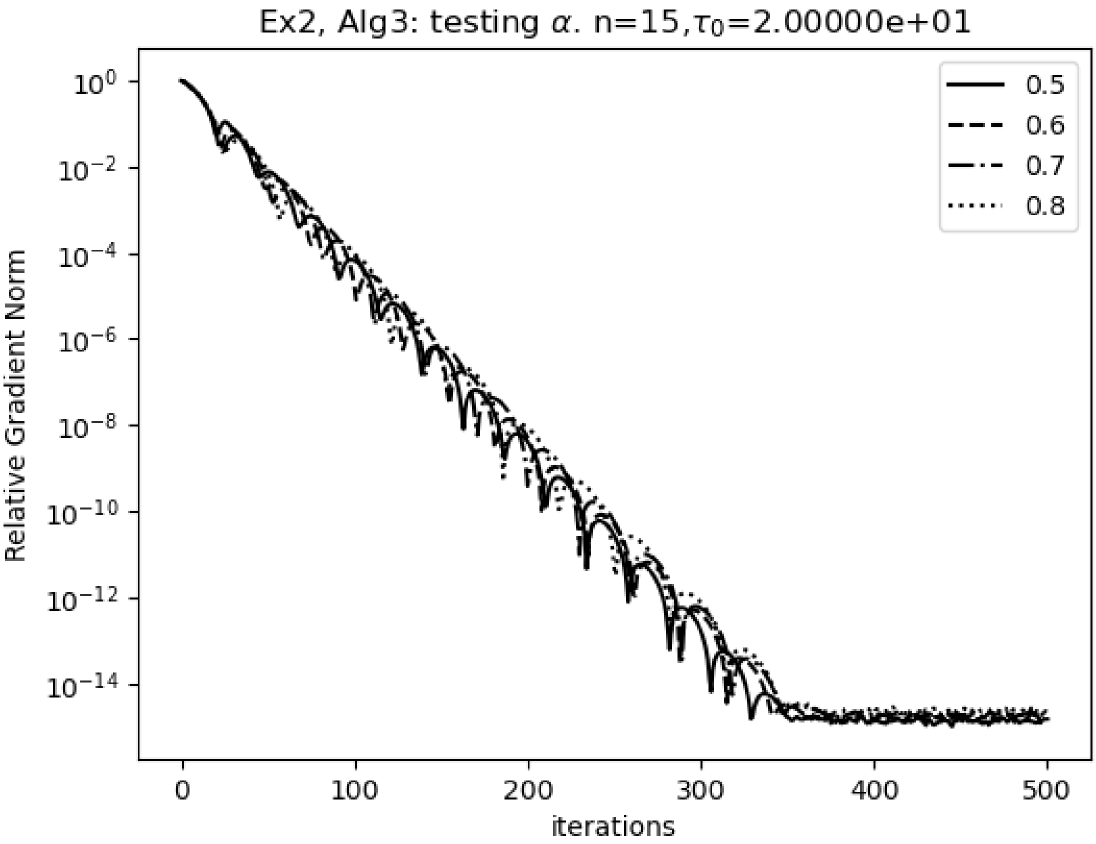


Figure 5d.

```
[41]: Figure5bd(15; tau0=20.0, pvec=[.5, .6, .7, .8], maxit=500);
```


Ex2, Alg3: testing α. n=15,τ_0=2.00000e+01
Relative Gradient Norm
10^{0}
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
10^{-14}
0
100
200
300
400
500
iterations
0.5
0.6
0.7
0.8

[ ]: