PS 7

Due: Mon, Apr 13

1: By the Book Section 9.4, Problem 6. For part (b), it may help to refer to a homework problem earlier in the semester!

2: Naive Newton Consider the nonlinear system of equations

$$\begin{aligned} x^2 + xy^2 &= 9 \\ 3x^2y - y^3 &= 4 \end{aligned}$$

Fill in the following Newton iteration code:

```
for k = 1:20
  F = % Your code here
  if norm(F) < rtol
    break;
  end
  J = % Your code here
  dx = J\F;
  x = x-dx;
end
```

Run your code with an initial guess of $(1, 1)$ and a residual norm tolerance of 10^{-12}. Do you see quadratic convergence?

3: Continue with Care Consider the boundary value problem

$$\begin{aligned} v''(x) + \gamma \exp(v(x)) = 0, &\quad 0 < x < 1 \\ v(0) = v(1) = 0 \end{aligned}$$

discretized via finite differences on a mesh with 100 equally spaced points; see example 9.3 in the book.

- Write a code to find v for a range of γ values from 1 to 3.5 (use `gammas = linspace(1,3.5)` to generate the mesh). For the first value of γ, you should use an initial guess of $v = 0$; for subsequent values, use the value of v at the previous γ step. Plot all your solutions together on a single plot.

- For all γ in the given range, the Jacobian matrix at the solution remains negative definite. Plot $\lambda_{\max}(J(v^*))$ (the eigenvalue closest to zero) as a function of γ. What do you notice?
- Try running your code again, this time going up to a maximum value of 4 rather than 3.5. What happens?

Note: You may start from the following code

```
n = 100;
h = 1/(n+1);
T = (n+1)^2 * (diag(ones(n-1,1),-1) + diag(one(n-1,1),1) - 2*eye(n));
v = zeros(n,1);
```

If n was very large, we might want to use a sparse matrix[1], but it's probably not worth it in this case.

[1] I'd probably switch to a more accurate discretization method, first, but that's a topic for CS 4210.