ASSIGNMENT 3 SOLUTIONS

1. A well-conditioned problem: interpolation in Chebyshev points

What is the smallest integer n for which this bound does not ensure $\|\Delta p\| \le 10\|\Delta f\|$?

The equation to solve is $1 + (2/\pi)\log(n+1) = 10$, i.e., $n = \exp(9\pi/2) - 1$, whose solution is

```
format short
n_real = exp(9*pi/2) - 1
```

```
n_real = 1.3794e+06
```

So the answer is

```
n = ceil(n_real)
```

```
n = 1379410
```

2. An unstable algorithm: polyval(polyfit)

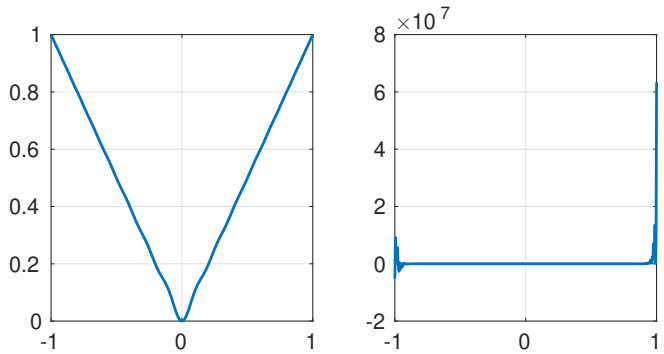
(a) Look up Matlab `polyfit` *and* `polyval` *and use them to compute Chebyshev interpolants of $f(x) = |x|$ on $[-1,1]$ for $n = 40$ and 80. What is the ∞-norm of the vector c? Use polyval to compute p(*`xx`*) for* `xx = linspace(-1,1,500)'` *and plot* `xx` *against p(*`xx`*).*

Note the Matlab warnings. This not very helpful message goes back decades.

```
xx = linspace(-1,1,500)';
for k = 1:2
    n = 40*k; subplot(1,2,k)
    x = cos((0:n)'*pi/n);
    c = polyfit(x,abs(x),n); normc = norm(c,inf)
    plot(xx,polyval(c,xx)), grid on
end
```

```
Warning: Polynomial is badly conditioned. Add points with distinct X values,
reduce the degree of the polynomial, or try centering and scaling as described
in HELP POLYFIT.
normc = 2.9374e+11

Warning: Polynomial is badly conditioned. Add points with distinct X values,
reduce the degree of the polynomial, or try centering and scaling as described
in HELP POLYFIT.
normc = 1.2884e+23
```



(b) Comment on how the norms you just calculated relate to the plots you just plotted in light of the value of machine epsilon of around 10^{-16}.

For $n = 60$ the coefficients c_k are of order 10^{23}, so in 16-digit arithmetic, computations involving them can be expected to introduce rounding errors on the order of 10^7. This is just what we saw.

(c) Calculate the condition numbers of A for $n = 40$ *and* 80. *Comment on their significance in the light of machine epsilon.*

The condition numbers come out as $O(10^{14})$ and $O(10^{18})$, respectively. Presumably the first number is mathematically accurate, but the second is surely an underestimate of the mathematical truth; there is no trusting anything bigger than $O(10^{16})$. But in any case its size much bigger than 10^{16} confirms that solutions to $Ac = f$ must be expected to have no useful accuracy.

```
n = 40; x = cos((0:n)*pi/n); cond40 = cond(vander(x))
n = 80; x = cos((0:n)*pi/n); cond80 = cond(vander(x))

cond40 = 7.7166e+14
cond80 = 5.0763e+18
```

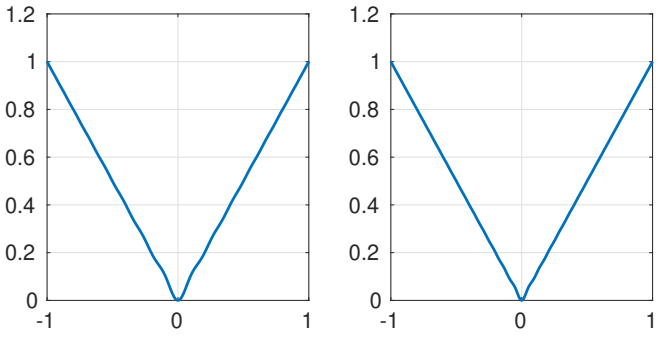
3. A stable algorithm: polyval(polyfit) + Arnoldi

Use these codes to compute the same interpolants for $n = 40$ *and* $n = 80$ *as in problem* 2. *Plot the results as before and give the* ∞*-norm of* c. *Comment on what you see.*

Now the norms of c are $O(1)$ and the computation is successful.

```
for k = 1:2
    n = 40*k; subplot(1,2,k)
    x = cos((0:n)'*pi/n);
    [c,H] = polyfitA(x,abs(x),n); normc = norm(c,inf)
    plot(xx,polyvalA(c,H,xx)), grid on
end

normc = 0.6452
normc = 0.6410
```



4. Least-squares on a more general domain

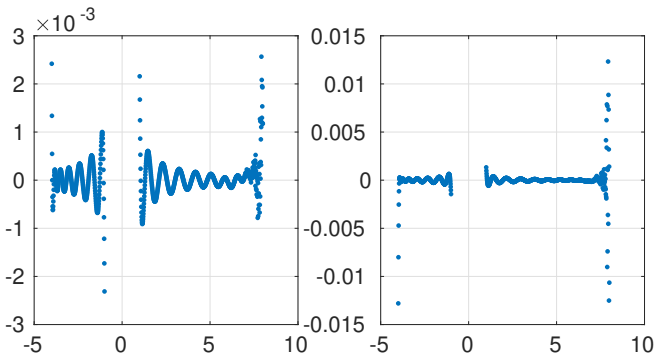
(a) Discretize X by a vector `xx` *with* 300 *equispaced points on the left and* 700 *on the right. Use* `polyfit` *and* `polyval` *to compute least-squares fits of degrees* $n = 40$ *and* 80*, and plot the error function in each case.*

Again, terribly unstable.

```
xx = [linspace(-4,-1,300) linspace(1,8,700)]';
for k = 1:2
    n = 40*k; subplot(1,2,k)
    c = polyfit(xx,sign(xx),n);
    plot(xx,sign(xx) - polyval(c,xx),'.'), grid on
end
```

```
Warning: Polynomial is badly conditioned. Add points with distinct X values,
reduce the degree of the polynomial, or try centering and scaling as described
in HELP POLYFIT.

Warning: Polynomial is badly conditioned. Add points with distinct X values,
reduce the degree of the polynomial, or try centering and scaling as described
in HELP POLYFIT.
```



(b) Likewise with `polyfitA` *and* `polyvalA`*. Comment on the difference*

Again, much better.

```
for k = 1:2
    n = 40*k; subplot(1,2,k)
    [c,H] = polyfitA(xx,sign(xx),n);
    plot(xx,sign(xx) - polyvalA(c,H,xx),'.'), grid on
end
```