HWfuncapp Questions

March 28, 2019

1 1. Plain Vanilla Chebyshev

In this exercise you should write some code that approximates the function $f(x) = x + 2x^2 - exp(-x)$ for $x \in [-3, 3]$. You should define a function `q1(n)`, where `n` is the number of interpolation points. You should use an approximation of degree `deg=n-1`, and you should set `n=15` Chebyshev interpolation nodes.

- predict `n_new=100` new equally spaced points in $[-3, 3]$ using your interpolator.
- make a plot with 2 panels. panel 1 shows true function values and your approximation and panel 2 shows the deviation in your approximation from the true f.
- Write an automated test that passes if the maximal deviation in your approximation from the true f is smaller than `1e-9`.

2 2. Question 1. with `ApproxFun.jl`

Redo exercise 1 with the `ApproxFun.jl` package. This should go into function `q2(n::Number)`, where now `n` is a positive number that will be used to define a symmetric interval around zero. So, you give `n=4`, you create $[-4, 4]$ with `ApproxFun`.

3 3. More fun with `ApproxFun.jl`

Let's use the same setup as in question 2, `q3(n::Number)`. Now however, we want to combine 2 functions into a third one:

$$f(x) = sin(x^2) g(x) = cos(x) h(x) = f(x) - g(x)$$

Use `ApproxFun.jl` to define this approximation. then get all the roots of h, ie. all $xs.t.h(x) = 0$. Make a plot of $h(x), x \in [-10, 10]$, and plot the roots of h onto the same plot. Finally, compute the definite integral

$$\int_{-10}^{0} h(z) dz$$

and return it's value.

Notice that by looking at the readme of that package, you will go pretty far here.

4 4. Plotting the Chebyshev Basis (Optional)

plot the first nine Chebyshev basis functions in function `q4()`. You could take some inspiration for the plot from yesterday's slides.

5 5. Importance of node placement: uniform vs chebyshev nodes (Optional)

In this exercise we want to investigate one of the dangers with polynomial approximation: placement of knots. We will focus on the classic example of Runge's function:

$$f(x) = \frac{1}{1+25x^2}, x \in [-5, 5]$$

Here we want to approximate f with the Chebyshev polynomial. We want to learn the impact of evaluating the polynomial at an equidistant set of points as opposed to something else.

- Produce a plot with 2 panels, *uniformly spaced* and *chebyshev nodes*:
  - Panel 1 should show approximations using a uniformly spaced grid of interpolation points.
  - Panel 2 should show approximations using chebyshev interpolation points.
- Each panel should show 4 lines:
  1. The true function
  2. the resulting approximations from a Chebyshev Polynomial Interpolation of degrees $k = 5, 9, 15$. Are we going to get a better approximation as we increase k?
- For each approximation, you should choose $n = k + 1$ interpolation points.
- The code contains a custom `ChebyType` that I found useful to produce these plots. You may want to use it as well.

6 6. Spline Interpolation: where to place knots? (Optional)

- Produce a plot with 2 panels. Panel 1 (*Runge's function*) shows $f(x), x \in [-5, 5]$, panel 2 (*Error in Runge's function*) shows deviations of your approximation to it (see below).
- Use 2 versions of a cubic spline to approximate the function:
  - version 1: use 13 equally spaced knots when you set up the `BSpline` object. Remember, that this is

```
using ApproXD
b = BSpline(nknots,deg,lb,ub)  # knot vector is chosen for you
```

  - version 2: use 13 knots that are concentrated towards 0.

```
using ApproXD
b = BSpline(my_knots,deg)   # you choose my_knots
```

- To estimate your approximating coefficients, evaluate the basis and f at `nevals=5*13=65` equidistant points.
- So, panel 2 of your plot should show 2 lines (`f - approx1` and `f - approx2`), as well as the placement of your concentrated knots.

7 7. Splines and Kinks

Interpolate the function $f(x) = |x|^{0.5}, x \in [-1, 1]$ with a cubic spline, with 13 knots, and 65 evaluation points to estimate it's coefficients.

Produce a plot with 3 panels:

1. Plot the true function. It has a kink at 0.
2. plot two spline approximations.
    1. a spline with a uniform knot vector.
    2. a spline with a knot vector that has a knot multiplicity at the kink $x = 0$. How many knots do you have to set equal to zero to produce a kink? note that the total number of (interior) knots should not change (i.e. 13).
3. In panel three plot the errors of both approximations.

```
In [ ]:
```