IntermediateProblemAnswers

November 12, 2018

1 Intermediate Problem Answers

1.1 MyRange and MyLinSpace Problem

1.1.1 Part 1

```
In [4]: struct MyRange
            start
            step
            stop
        end

        function _MyRange(a::MyRange,i::Int)
            tmp = a.start + a.step*(i-1)
            if tmp > a.stop
                error("Index is out of bounds!")
            else
                return tmp
            end
        end
```

```
Out[4]: _MyRange (generic function with 1 method)
```

```
In [5]: a = MyRange(1,2,20)
        _MyRange(a,5) == (1:2:20)[5]
```

```
Out[5]: true
```

```
In [6]: Base.getindex(a::MyRange,i::Int) = _MyRange(a,i)
        a[5]
```

```
Out[6]: 9
```

1.1.2 Part 2

```
In [7]: struct MyLinSpace
            start
            stop
            n
```

```
        end

        function Base.getindex(a::MyLinSpace,i::Int)
            dx = (a.stop-a.start)/a.n
            a.start + dx*(i-1)
        end
```

```
In [8]: l = MyLinSpace(1,2,50)
        l[6]
```

```
Out[8]: 1.1
```

```
In [10]: range(1,stop=2,length=50)[6]
```

```
Out[10]: 1.1020408163265305
```

1.1.3 Part 3

```
In [11]: (a::MyRange)(x) = a.start + a.step*(x-1)
         a = MyRange(1,2,20)
         a(1.1)
```

```
Out[11]: 1.2000000000000002
```

1.1.4 Part 4

```
In [12]: using Unitful
         a = MyRange(1u"kg",2u"kg",20u"kg")
         a[5]
```

```
Out[12]: 9 kg
```

1.2 Operator Problem

```
In [19]: struct StrangMatrix end
         A = StrangMatrix()

         using LinearAlgebra
         function LinearAlgebra.mul!(C,A::StrangMatrix,B::AbstractVector)
             for i in 2:length(B)-1
                 C[i] = B[i-1] - 2B[i] + B[i+1]
             end
             C[1] = -2B[1] + B[2]
             C[end] = B[end-1] - 2B[end]
             C
         end

         Base.:*(A::StrangMatrix,B::AbstractVector) = (C = similar(B); mul!(C,A,B))
```

```
In [20]: A*ones(10)
```

```
Out[20]: 10-element Array{Float64,1}:
          -1.0
           0.0
           0.0
           0.0
           0.0
           0.0
           0.0
           0.0
           0.0
          -1.0
```

```
In [21]: struct SizedStrangMatrix
             size
         end

         Base.eltype(A::SizedStrangMatrix) = Float64
         Base.size(A::SizedStrangMatrix) = A.size
         Base.size(A::SizedStrangMatrix,i::Int) = A.size[i]
```

```
In [22]: b = sin.(0:0.1:2)
```

```
Out[22]: 63-element Array{Float64,1}:
           0.0
           0.09983341664682815
           0.19866933079506122
           0.2955202066613396
           0.3894183423086505
           0.479425538604203
           0.5646424733950355
           0.6442176872376911
           0.7173560908995228
           0.7833269096274834
           0.8414709848078965
           0.8912073600614354
           0.9320390859672264

          -0.9258146823277321
          -0.8834546557201531
          -0.8322674422239008
          -0.7727644875559871
          -0.7055403255703919
          -0.6312666378723208
          -0.5506855425976376
          -0.4646021794137566
          -0.373876664830236
          -0.27941549819892586
          -0.18216250427209502
          -0.0830894028174964
```

```
In [24]: A = SizedStrangMatrix((length(b),length(b)))

         function LinearAlgebra.mul!(C,A::SizedStrangMatrix,B)
             for i in 2:length(B)-1
                 C[i] = B[i-1] - 2B[i] + B[i+1]
             end
             C[1] = -2B[1] + B[2]
             C[end] = B[end-1] - 2B[end]
             C
         end
         Base.:*(A::SizedStrangMatrix,B::AbstractVector) = (C = similar(B); mul!(C,A,B))
```

```
In [29]: using IterativeSolvers
         x = gmres(A,b,tol=1e-14)
```

```
Out[29]: 63-element Array{Float64,1}:
            -9.755738784527166
           -19.511766690859723
           -29.168509953717926
           -38.62733151652308
           -47.79172939964289
           -56.567920000176755
           -64.86624757504093
           -72.60180867498484
           -79.69549984767762
           -86.07440664159829
           -91.67293002909796
           -96.43316292771152
          -100.30568365156476

            92.73105600644514
            88.6304815198612
            83.64512976504756
            77.82628337498842
            71.23343662444809
            63.93401241944272
            56.00167853409517
            47.51925322977693
            38.572331686261165
            29.25162537589525
            19.65105899819377
             9.867868177542748
```

```
In [30]: A*x - b
```

```
Out[30]: 63-element Array{Float64,1}:
          -0.00028912180539109045
          -0.0005487731724762024
          -0.0007476307420101191
```

```
 -0.0010965269760021545
 -0.0012110597227052433
 -0.0015625129345115951
 -0.0018759984747689362
 -0.00234775998656922
 -0.0025717121273999277
 -0.002943503206488285
 -0.0031804959217944795
 -0.00349518530111137
 -0.00378335609624314

 -0.001396616777037063
 -0.0013226125095587848
 -0.001227193021585582
 -0.0012358729252088807
 -0.0010371288946419144
 -0.0016430424698667956
  0.0005941236269499939
  0.00010594021623389072
  9.189798008585326e-5
 -0.0004445691366434801
 -0.000461938677442153
 -0.001587954074231443
```

1.3 Regression Problem

```
In [33]: #### Prepare Data

         X = rand(1000, 3)               # feature matrix
         a0 = rand(3)                    # ground truths
         y = X * a0 + 0.1 * randn(1000);  # generate response

         X2 = hcat(X,ones(1000))
         println(X2\y)

         using MultivariateStats
         println(llsq(X,y))

         using DataFrames, GLM
         data = DataFrame(X1=X[:,1], X2=X[:,2], X3=X[:,3],Y=y)
         OLS = lm(@formula(Y ~ X1 + X2 + X3), data)


         X = rand(100);
         y = 2X  + 0.1 * randn(100);

         using Plots
         b = X\y
```

```
        println(b)
        gr()
        scatter(X,y)
        Plots.abline!(b[1],0.0, lw=3) # Slope,Intercept
```

```
[0.400269, 0.834389, 0.0869948, -0.00661511]
```

```
 Info: Precompiling MultivariateStats [6f286f6a-111f-5878-ab1e-185364afe411]
 @ Base loading.jl:1186
ERROR: LoadError: UndefVarError: LinAlg not defined
Stacktrace:
 [1] include at ./boot.jl:317 [inlined]
 [2] include_relative(::Module, ::String) at ./loading.jl:1038
 [3] include(::Module, ::String) at ./sysimg.jl:29
 [4] top-level scope at none:2
 [5] eval at ./boot.jl:319 [inlined]
 [6] eval(::Expr) at ./client.jl:389
 [7] top-level scope at ./none:3
in expression starting at /home/chrisrackauckas/.julia/packages/MultivariateStats/wGpiN/src/Mul
```

```
        Failed to precompile MultivariateStats [6f286f6a-111f-5878-ab1e-185364afe411] to /home/

        Stacktrace:

         [1] error(::String) at ./error.jl:33

         [2] macro expansion at ./logging.jl:313 [inlined]

         [3] compilecache(::Base.PkgId, ::String) at ./loading.jl:1184

         [4] macro expansion at ./logging.jl:311 [inlined]

         [5] _require(::Base.PkgId) at ./loading.jl:941

         [6] require(::Base.PkgId) at ./loading.jl:852

         [7] macro expansion at ./logging.jl:311 [inlined]

         [8] require(::Module, ::Symbol) at ./loading.jl:834

         [9] top-level scope at In[33]:10
```

1.4 Type Hierarchy Problem

```
In [34]: abstract type AbstractPerson end
         abstract type AbstractStudent <: AbstractPerson end

         struct Person <: AbstractPerson
             name
         end

         struct Student <: AbstractStudent
             name
             grade
         end

         struct GraduateStudent <: AbstractStudent
             name
             grade
         end

         person_info(p::AbstractPerson) = println(p.name)
         person_info(s::AbstractStudent) = (println(s.name); println(s.grade))

Out[34]: person_info (generic function with 2 methods)

In [35]: person_info(Person("Bob"))

Bob


In [36]: person_info(Student("Bob",2))

Bob
2


In [37]: person_info(GraduateStudent("Bob",2))

Bob
2
```

1.5 Distribution Dispatch Problem

This is from Josh Day's talk: https://www.youtube.com/watch?v=EwcTNzpQ6Sc
Solution is from: https://github.com/joshday/Talks/blob/master/SLG2016_IntroToJulia/Slides.ipynb

```
In [39]: using Distributions
         function myquantile(d::UnivariateDistribution, q::Number)
              = mean(d)
             tol = Inf
```

```
    while tol > 1e-5
        old =
         =  - (cdf(d, ) - q) / pdf(d, )
        tol = abs(old - )
    end

end

for dist in [Gamma(5, 1), Normal(0, 1), Beta(2, 4)]
    @show myquantile(dist, .75)
    @show quantile(dist, .75)
    println()
end
```

```
myquantile(dist, 0.75) = 6.274430698436519
quantile(dist, 0.75) = 6.2744306984446885

myquantile(dist, 0.75) = 0.6744897501960708
quantile(dist, 0.75) = 0.6744897501960818

myquantile(dist, 0.75) = 0.45418056477357555
quantile(dist, 0.75) = 0.4541805647736157
```