ProblemAnswers

October 24, 2016

0.1   Problem 1

```
In [10]: N = 10
         A = zeros(N,N)
         for i in 1:N, j in 1:N
             abs(i-j)<=1 ? A[i,j]+=1 : nothing
             i==j ? A[i,j]-=3 : nothing
         end
         A
```

```
Out[10]: 10×10 Array{Float64,2}:
         -2.0    1.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
          1.0   -2.0    1.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0
          0.0    1.0   -2.0    1.0    0.0    0.0    0.0    0.0    0.0    0.0
          0.0    0.0    1.0   -2.0    1.0    0.0    0.0    0.0    0.0    0.0
          0.0    0.0    0.0    1.0   -2.0    1.0    0.0    0.0    0.0    0.0
          0.0    0.0    0.0    0.0    1.0   -2.0    1.0    0.0    0.0    0.0
          0.0    0.0    0.0    0.0    0.0    1.0   -2.0    1.0    0.0    0.0
          0.0    0.0    0.0    0.0    0.0    0.0    1.0   -2.0    1.0    0.0
          0.0    0.0    0.0    0.0    0.0    0.0    0.0    1.0   -2.0    1.0
          0.0    0.0    0.0    0.0    0.0    0.0    0.0    0.0    1.0   -2.0
```

```
In [17]: #### Prepare Data

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

```
Out[17]: 1000-element Array{Float64,1}:
          0.245515
          0.213645
          0.179037
         -0.0252645
          0.327495
          0.1073
          0.284771
          0.134653
          0.32761
```

```
       0.139108
       0.321555
       0.192148
       0.124061
          ⋮
       0.173127
       0.203367
       0.183517
       0.107017
       0.290129
       0.203007
       0.261345
       0.205973
       0.194457
       0.0825708
       0.180052
       0.0156171
```

0.2   Problem 2

```
In [18]: X2 = hcat(X,ones(1000))
         println(X2\y)

[0.219939,0.0336012,0.090267,0.00117261]
```

0.3   Problem 3

```
In [19]: using MultivariateStats
         llsq(X,y)

Out[19]: 4-element Array{Float64,1}:
          0.219939
          0.0336012
          0.090267
          0.00117261
```

0.4   Problem 4

```
In [20]: using DataFrames, GLM
         data = DataFrame(X1=X[:,1], X2=X[:,2], X3=X[:,3],Y=y)
         OLS = lm(Y ~ X1 + X2 + X3, data)

Out[20]: DataFrames.DataFrameRegressionModel{GLM.LinearModel{GLM.LmResp{Array{Float

         Formula: Y ~ 1 + X1 + X2 + X3

         Coefficients:
                      Estimate Std.Error  t value Pr(>|t|)
```

```
(Intercept)   0.00117261   0.010462 0.112083    0.9108
X1                 0.219939 0.0113648   19.3526   <1e-70
X2                 0.0336012 0.0111465   3.01451   0.0026
X3                 0.090267 0.0108559    8.315    <1e-15
```

0.5 Problem 5

```
In [21]: r = 2.9:.00005:4; numAttract = 150
         steady = ones(length(r),1)*.25
         for i=1:400 ## Get to steady state
           steady .= r.*steady.*(1-steady)
         end
         x = zeros(length(steady),numAttract)
         x[:,1] = steady
         @inbounds for i=2:numAttract ## Grab values at the attractor
           x[:,i] = r.*x[:,i-1].*(1-x[:,i-1])
         end
         using Plots
         plot(collect(r),x,seriestype=:scatter,markersize=.002,legend=false)
```

0.6 Metaprogramming Project

```
In [10]: macro ~(y,ex)
            new_ex = Meta.quot(ex)
            quote
              inner_ex = $(esc(new_ex))
              data_name = symbol(string(inner_ex.args[end])[1])
              eval_ex = Expr(:(=),:data,data_name)
              eval(Main,eval_ex)
              new_X = Matrix{Float64}(size(data,1),length(inner_ex.args)-1)
              cur_spot = 0
              for i in 2:length(inner_ex.args)
                if inner_ex.args[i] == 1
                  new_X[:,i-1] = ones(size(data,1))
                else
                  col = parse(Int,string(string(inner_ex.args[i])[2]))
                  new_X[:,i-1] = data[:,col]
                end
              end
              $(esc(y)),new_X
            end
         end

         y = rand(10)
         X = rand(10,4)
         y~1+X1+X2+X4
```

```julia
function solve_least_squares(y,X)
    X\y
end
solve_least_squares(tup::Tuple) = solve_least_squares(tup...)
solve_least_squares(y~1+X1+X2+X4)
```

WARNING: Method definition @~(

Out[10]: 4-element Array{Float64,1}:
         0.529064
         0.55191
        -0.52334
         0.0513438

ANY<:Any, ANY<:Any) in module Main at In[9]:2 overwritten at In[10]:2.
WARNING: Method definition solve_least_squares(Any, Any) in module Main at In[4]:27
WARNING: Method definition solve_least_squares(Tuple) in module Main at In[4]:29 ov