BasicProblemAnswers

May 26, 2017

1 Starter Problems

1.1 Strang Matrix Problem

```
In [3]: N = 10
        A = zeros(N,N)
        for i in 1:N, j in 1:N
            abs(i-j)<=1 && (A[i,j]+=1)
            i==j && (A[i,j]-=3)
        end
        A
```

```
Out[3]: 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
```

1.2 Factorial Problem

```
In [1]: function my_factorial(n)
            k = one(n)
            for i in 1:n
                k *= i
            end
            k
        end

        my_factorial(4)
        my_factorial(30)
        my_factorial(big(30))
```

```
Out[1]: 265252859812191058636308480000000
```

1.3 Binomial Problem

```
In [1]: function binomial_rv(n, p)
            count = zero(n)
            U = rand(n)
            for i in 1:n
              U[i] < p && (count += 1)
            end
            count
        end

        bs = [binomial_rv(10, 0.5) for j in 1:10]
```

```
Out[1]: 10-element Array{Int64,1}:
         1
         5
         6
         6
         2
         4
         3
         5
         5
         8
```

1.4 Monte Carlo π Problem

```
In [17]: n = 10000000

         count = 0
         for i in 1:n
             u, v = 2rand(2)-1
             d = sqrt(u^2 + v^2)  # Distance from middle of square
             d < 1 && (count += 1)
         end

         area_estimate = count / n

         print(area_estimate * 4)  # dividing by radius**2
```

```
3.1417656
```

2 Integration Problems

2.1 Timeseries Generation Problem

```
In [22]: alphas = [0.0, 0.5, 0.98]
         T = 200
```

```
series = []
labels = []

for alpha in alphas
    x = zeros(T + 1)
    x[1] = 0.0
    for t in 1:T
        x[t+1] = alpha * x[t] + randn()
    end
    push!(series, x)
    push!(labels, "alpha = $alpha")
end

plot(series, label=reshape(labels,1,length(labels)),lw=3)
```

2.2 Linear Regression Problem

In [2]:
```
#### 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
```

```
[0.474905,0.910809,0.298824,0.00130641]
[0.474905,0.910809,0.298824,0.00130641]
[1.9975]
```

```
WARNING: Method definition describe(AbstractArray) in module StatsBase at /home/cra
```

2.3 Logistic Equation Problem

```
In [9]: r = 2.9:.001:4; numAttract = 100
        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; gr()
        plot(collect(r),x,seriestype=:scatter,markersize=.002,legend=false,color=:k
```