Pset3_JuliaCode

April 5, 2017

```
In [2]: using JuMP, JuMPeR, Gurobi, PyPlot, Distributions
```

```
In [3]: function RO_haveFacilities(n,p,c,μ,σ,t)
            ROO = RobustModel(solver=GurobiSolver(OutputFlag=0))
            @uncertain(ROO, u[1:n])
            @uncertain(ROO, d[1:n])
            @variable(ROO, x[1:n]>=0)
            @variable(ROO, s[1:n]>=0)
            @variable(ROO, y_minus[1:n,1:n]<=0)

            @constraint(ROO, norm(u,1)<=n^0.5)
            @constraint(ROO, norm(u,Inf)<=1)
            @constraint(ROO, d .== μ+σ.*u)

            for i = 1:n
                @constraint(ROO, s[i]<=d[i])
                @constraint(ROO, s[i]<=x[i]+sum(y_minus[i,j] for j=1:n)-sum(y_minus[j,i] for j=1:n))
            end

            @objective(ROO, Max, p*sum(s)-sum(x)+sum(sum(t[i,j]*y_minus[i,j] for i=1:n ) for j=1:n))
            solve(ROO)
            return getvalue(x),getvalue(y_minus)
        end
```

```
Out[3]: RO_haveFacilities (generic function with 1 method)
```

```
In [4]: function AAO_haveFacilities(n,p,c,μ,σ,t)
            AAOO = RobustModel(solver=GurobiSolver(OutputFlag=0))
            @uncertain(AAOO, u[1:n])
            @uncertain(AAOO, d[1:n])
            @variable(AAOO, x[1:n]>=0)
            @variable(AAOO, s[1:n]>=0)
            @variable(AAOO, F)
            @adaptive(AAOO, y_minus[i=1:n,j=1:n]<=0, policy=Affine, depends_on=d[1:n])
            @constraint(AAOO, norm(u,1)<=n^0.5)
            @constraint(AAOO, norm(u,Inf)<=1)
            @constraint(AAOO, d .== μ+σ.*u)

            for i = 1:n
                @constraint(AAOO, s[i]<=d[i])
                @constraint(AAOO, s[i]<=x[i]+sum(y_minus[i,j] for j=1:n)-sum(y_minus[j,i] for j=1:n))
            end
             @constraint(AAOO, p*sum(s)-sum(x)+sum(sum(t[i,j]*y_minus[i,j] for i=1:n ) for j=1:n)>=F)
             @objective(AAOO, Max, F)
            solve(AAOO)
            return getvalue(x)
        end
```

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

```
In [5]: function Optimal_haveFacilities(n,p,c,μ,σ,t,d,x)
            OptO = Model(solver=GurobiSolver(OutputFlag=0))
            @variable(OptO, y_minus[1:n,1:n]<=0)
            @variable(OptO, s[1:n]>=0)
            @variable(OptO,F)
            for i = 1:n
                @constraint(OptO, s[i]<=d[i])
                @constraint(OptO, s[i]<=x[i]+sum(y_minus[i,j] for j=1:n)-sum(y_minus[j,i] for j=1:n))
            end
            @constraint(OptO, p*sum(s)-sum(x)+sum(sum(t[i,j]*y_minus[i,j] for i=1:n ) for j=1:n)>=F)
            @objective(OptO, Max, F)
            solve(OptO)
            return getobjectivevalue(OptO)
        end
```

```
Out[5]: Optimal_haveFacilities (generic function with 1 method)
```

```
In [6]: function ObjVal_haveFacilities(n,p,c,x,y_minus,d)
            s = zeros(n,1)
            for i = 1:n
                s[i]=min(d[i],x[i]+sum(y_minus[i,:])-sum(y_minus[:,i]))
            end
            p*sum(s)-sum(x)+sum(sum(t.*y_minus))
        end
```

```
Out[6]: ObjVal_haveFacilities (generic function with 1 method)
```

```
In [7]: for n in 5:5:25

            err = 0
            for r = 1:5
                p = 3
                c = 30
                μ = rand(Uniform(50,150),n)
                σ = 0.5 * μ

                t = zeros(n,n)
                x_cor = rand(Uniform(0,1),n)
                y_cor = rand(Uniform(0,1),n)
                for i=1:n
                    for j=1:n
                        t[i,j] = ((x_cor[i]-x_cor[j])^2 + (y_cor[i]-y_cor[j])^2)^0.5
                    end
                end
                d = []
                for i = 1:n
                    push!(d , rand(Normal(μ[i],σ[i])))
                end
```

```
        RO_x,RO_y = RO_haveFacilities(n,p,c,μ,σ,t)
        AAO_x = AAO_haveFacilities(n,p,c,μ,σ,t)
        ROObj = ObjVal_haveFacilities(n,p,c,RO_x,RO_y,d)
        AAOObj = Optimal_haveFacilities(n,p,c,μ,σ,t,d,AAO_x)
        #print("n=",n,":",(AAOObj-ROObj)/AAOObj,"\n")

    end

end
```

```
UndefVarError: t not defined



 in ObjVal_haveFacilities(::Int64, ::Int64, ::Int64, ::Array{Float64,1}, ::Array{Float64,2}, ::A

 in macro expansion; at .\In[7]:24 [inlined]

 in anonymous at .\<missing>:?
```

In [8]:
```
function RO_noFacilities(n,p,c,μ,σ,t)
    M = 1000
    RO1 = RobustModel(solver=GurobiSolver(OutputFlag=0))
    @uncertain(RO1, u[1:n])
    @uncertain(RO1, d[1:n])
    @variable(RO1, x[1:n]>=0)
    @variable(RO1, s[1:n]>=0)
    @variable(RO1, y_minus[1:n,1:n]<=0)
    @variable(RO1, b[1:n], Bin)

    @constraint(RO1, norm(u,1)<=n^0.5)
    @constraint(RO1, norm(u,Inf)<=1)
    @constraint(RO1, d .== μ+σ.*u)

    for i = 1:n
        @constraint(RO1, s[i]<=d[i])
        @constraint(RO1, s[i]<=x[i]+sum(y_minus[i,j] for j=1:n)-sum(y_minus[j,i] for j=1:n))
        @constraint(RO1, x[i]<=M*b[i])
    end

    @objective(RO1, Max, p*sum(s)-sum(x)+sum(sum(t[i,j]*y_minus[i,j] for i=1:n ) for j=1:n)-c*su
    solve(RO1)
    return getvalue(x),getvalue(y_minus),getvalue(b)
end
```

Out[8]:
```
RO_noFacilities (generic function with 1 method)
```

In [9]:
```
function AAO_noFacilities(n,p,c,μ,σ,t)
    M = 1000
    AAO1 = RobustModel(solver=GurobiSolver(OutputFlag=0))
    @uncertain(AAO1, u[1:n])
    @uncertain(AAO1, d[1:n])
    @variable(AAO1, x[1:n]>=0)
```

```
            @variable(AAO1, s[1:n]>=0)
            @variable(AAO1, F)
            @variable(AAO1, b[1:n], Bin)
            @adaptive(AAO1, y_minus[i=1:n,j=1:n]<=0, policy=Affine, depends_on=d[1:n])
            @constraint(AAO1, norm(u,1)<=n^0.5)
            @constraint(AAO1, norm(u,Inf)<=1)
            @constraint(AAO1, d .== μ+σ.*u)

            for i = 1:n
                @constraint(AAO1, s[i]<=d[i])
                @constraint(AAO1, s[i]<=x[i]+sum(y_minus[i,j] for j=1:n)-sum(y_minus[j,i] for j=1:n))
                @constraint(AAO1, x[i]<=M*b[i])
            end
             @constraint(AAO1, -c*sum(b)+p*sum(s)-sum(x)+sum(sum(t[i,j]*y_minus[i,j] for i=1:n ) for j=

             @objective(AAO1, Max, F)
            solve(AAO1)
            return getvalue(x),getvalue(b)
        end

Out[9]: AAO_noFacilities (generic function with 1 method)

In [10]: function Optimal_noFacilities(n,p,c,μ,σ,t,d,x,b)
             Opt1 = Model(solver=GurobiSolver(OutputFlag=0))
             @variable(Opt1, y_minus[1:n,1:n]<=0)
             @variable(Opt1, s[1:n]>=0)
             @variable(Opt1,F)
             for i = 1:n
                 @constraint(Opt1, s[i]<=d[i])
                 @constraint(Opt1, s[i]<=x[i]+sum(y_minus[i,j] for j=1:n)-sum(y_minus[j,i] for j=1:n))
             end
             @constraint(Opt1, -c*sum(b)+p*sum(s)-sum(x)+sum(sum(t[i,j]*y_minus[i,j] for i=1:n ) for j=
             @objective(Opt1, Max, F)
             solve(Opt1)
             return getobjectivevalue(Opt1)
         end

Out[10]: Optimal_noFacilities (generic function with 1 method)

In [11]: function ObjVal_noFacilities(n,p,c,t,x,y_minus,d,b)
             s = zeros(n,1)
             for i = 1:n
                 s[i]=min(d[i],x[i]+sum(y_minus[i,:])-sum(y_minus[:,i]))
             end
             -c*sum(b)+p*sum(s)-sum(x)+sum(sum(t.*y_minus))
         end

Out[11]: ObjVal_noFacilities (generic function with 1 method)

In [12]: for n in 5:5:20

             err = 0
             for r = 1:1
                 p = 3
                 c = 30
                 μ = rand(Uniform(50,150),n)
                 σ = 0.5 * μ

                 t = zeros(n,n)
                 x_cor = rand(Uniform(0,1),n)
                 y_cor = rand(Uniform(0,1),n)
                 for i=1:n
                     for j=1:n
                         t[i,j] = ((x_cor[i]-x_cor[j])^2 + (y_cor[i]-y_cor[j])^2)^0.5
                     end
                 end
                 d = []
                 for i = 1:n
                     push!(d , rand(Normal(μ[i],σ[i])))
                 end
                 RO_x,RO_y,RO_b = RO_noFacilities(n,p,c,μ,σ,t)
                 AAO_x,AOO_b = AAO_noFacilities(n,p,c,μ,σ,t)
                 ROObj = ObjVal_noFacilities(n,p,c,t,RO_x,RO_y,d,RO_b)
                 AAOObj = Optimal_noFacilities(n,p,c,μ,σ,t,d,AAO_x,AOO_b)
                 #print("n=",n,":",(AAOObj-ROObj)/AAOObj,"\n")

            end

         end
```

```
WARNING: Not solved to optimality, status: Infeasible
```

In [ ]:

In [ ]:

In [ ]: