In [189]:
```
using Hyperopt,GalacticOptim
```

First, we define the optimization problem as:

In [161]:
```
rosenbrock(x,a) =  sum(@.  x + (a[1]-1)^2 + (a[2]-1)^2)
```

Out[161]:
```
rosenbrock (generic function with 2 methods)
```

Then, we follow the form like the example of GalacticOptim to wrap the function into an OptimizationProblem

In [162]:
```
x0 = zeros(1)
a0 = [1.00,1.00]
prob = OptimizationProblem(rosenbrock,x0,a0)
```

Out[162]:
```
OptimizationProblem. In-place: true
u0: [0.0]
```

Define a function as an API to searve for GalacticOptim. For "hyperparameters" optimization

In [182]:
```
"Define a function as an API to searve for GalacticOptim as a hyperparameters optimization
number of iteration. x is a variable that can be set in advance."
function Hyperoptimize(x,item)
ho = @hyperopt for i=item,
                    sampler = RandomSampler(), # This is default if none provided
                    a0[1] = LinRange(-2,2,1000),
                    #a0[2] = exp10.(LinRange(-2,2,1000))
                    a0[2] = LinRange(-2,2,1000)
                    print(i, "\t", a0[1], "\t", a0[2], "   \t")
                    x1 = x
                    @show prob.f(x1,a0)
                end
print('\n')
print("Minimize Parameters:")
print('\n')
printmin(ho)

end
```

Out[182]:
```
Hyperoptimize
```

In [154]:
```
? Hyperoptimize()
```

Out[154]:
Define a function as an API to searve for GalacticOptim as a hyperparameters optimization method. item is the parameter to control the number of iteration. x is a variable that can be set in advance.

We use the Hyperoptimize(x,item) to get our result

In [183]:
```
Hyperoptimize(10,600)
```

```
93541
594.0    0.47447447447447466     -0.6066066066066068     prob.f(x1, a0) = 12.857361866
370876
595.0    -0.35835835835835816    1.3713713713713713      prob.f(x1, a0) = 11.983054125
196267
596.0    -0.13013013013013008    -1.7197197197197198     prob.f(x1, a0) = 18.674069464
860256
597.0    1.6156156156156154      -0.8748748748748749     prob.f(x1, a0) = 13.894138382
62687
598.0    -1.2272272272272273     -0.09009009009009006    prob.f(x1, a0) = 16.148837526
214905
599.0    -1.3033033033033035     0.8988988988988988      prob.f(x1, a0) = 15.315427539
651763
600.0    -0.9109109109109107     -1.8878878878878878     prob.f(x1, a0) = 21.991476962
447933

Minimize Parameters:
a0[1] = 1.031031031031031
a0[2] = 0.9629629629629628
```

If we incorporate the following code into Galactic.jl/src/solve.jl function, we can use

sol = solve(prob,Hyperoptimize(10,10)) to get the final result.

In [ ]:
```
function Hyperoptimize(x,item)
ho = @hyperopt for i=item,
                    sampler = RandomSampler(), # This is default if none provided
                    a0[1] = LinRange(-2,2,1000),
                    #a0[2] = exp10.(LinRange(-2,2,1000))
                    a0[2] = LinRange(-2,2,1000)
                    print(i, "\t", a0[1], "\t", a0[2], "   \t")
                    x1 = x
                    @show prob.f(x1,a0)
                end
printmin(ho)
end


function __solve(prob::OptimizationProblem, opt::Hyperoptimize,
                 data = DEFAULT_DATA;
                 maxiters = nothing,
                 cb = (args...) -> (false),
                 progress = false,
                 kwargs...)
        Hyperoptimize(x,item)
end
```