In [20]:

```
Jclust(x,reps,colorac) = avg( [norm(x[i]-reps[colorac[i]])^2 for i=1:length(x)])
```

Out[20]:

```
Jclust (generic function with 1 method)
```

In [78]:

```
x = [ [0,1], [1,0], [-1,1],[0.9,-0.9],[1,-1],[1.1,1.07],[0.51,-0.23],[0.9,-0.8]]
```

Out[78]:

```
8-element Array{Array{Float64,1},1}:
 [0.0, 1.0]
 [1.0, 0.0]
 [-1.0, 1.0]
 [0.9, -0.9]
 [1.0, -1.0]
 [1.1, 1.07]
 [0.51, -0.23]
 [0.9, -0.8]
```

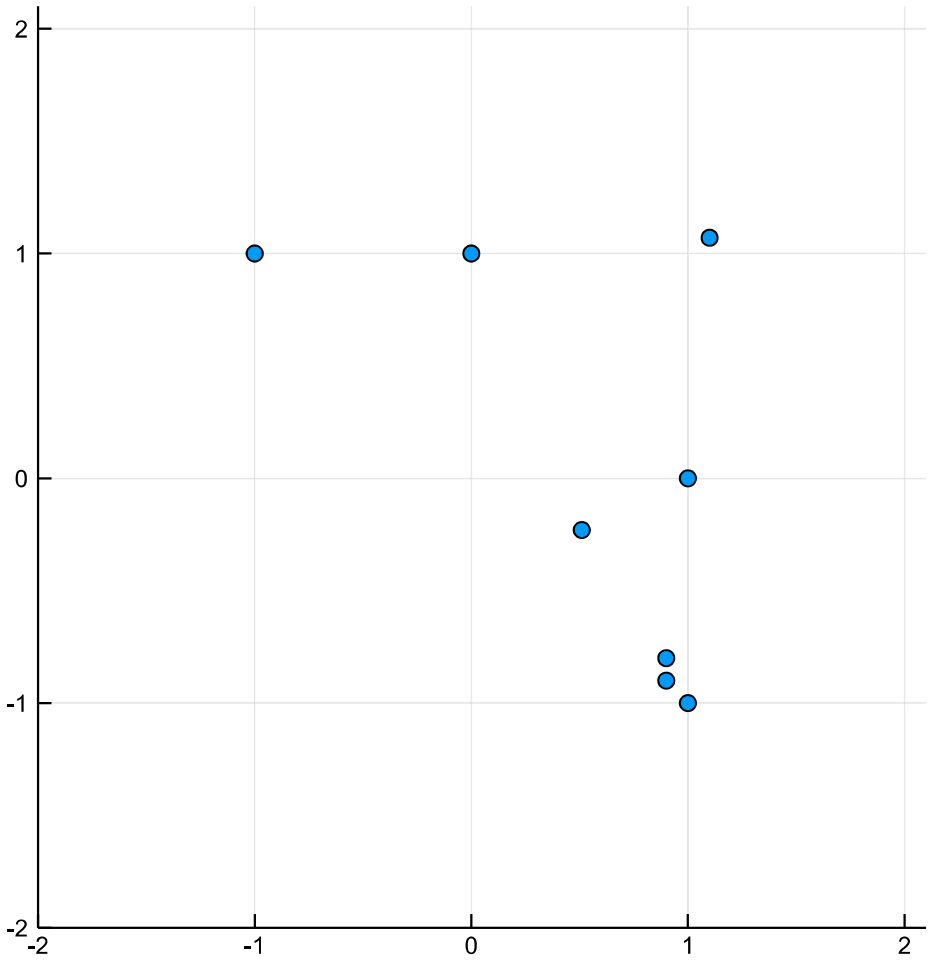
In [79]:

```
using Plots
```

In [80]:

```
scatter([c[1] for c in x], [c[2] for c in x])
plot!(lims = (-2, 2.1), size = (500,500), legend = false)
#savefig("dataset.pdf")
```

Out[80]:



In [81]:

```
reps = [ [1,1], [0,0],[0,1] ]
```

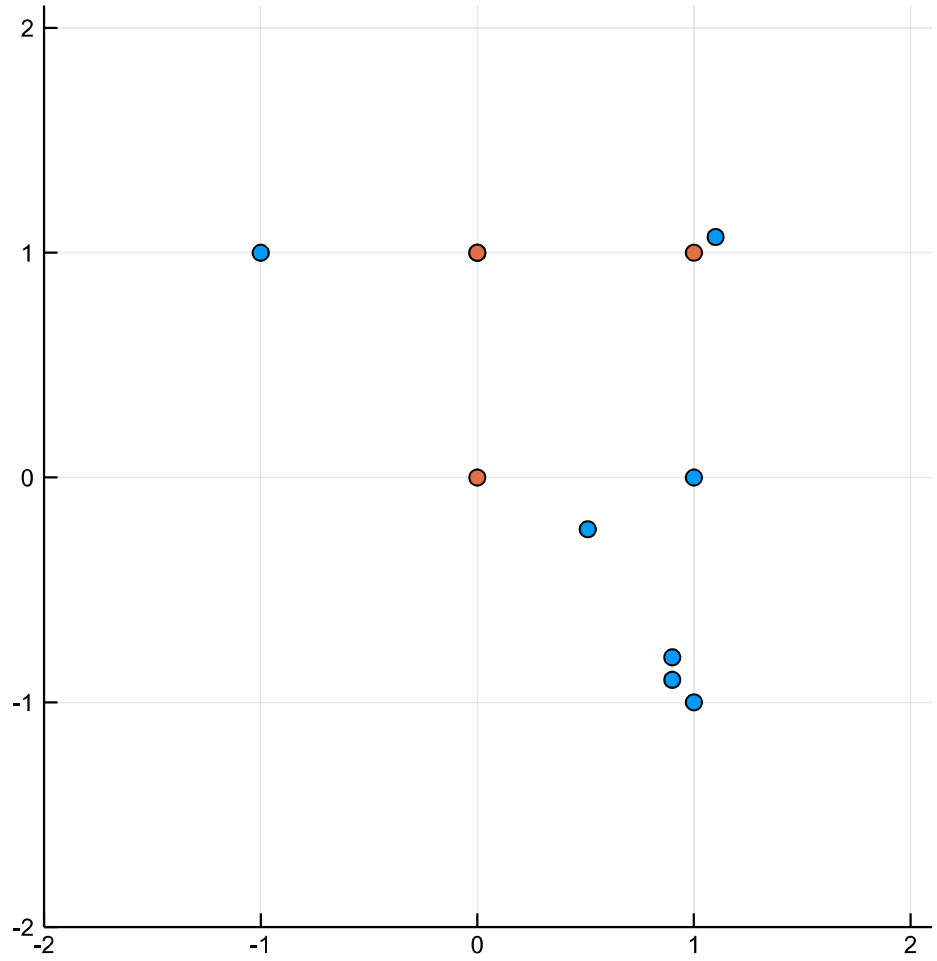
Out[81]:

```
3-element Array{Array{Int64,1},1}:
 [1, 1]
 [0, 0]
 [0, 1]
```

In [82]:

```
scatter([c[1] for c in x], [c[2] for c in x])
scatter!([c[1] for c in reps], [c[2] for c in reps])
plot!(lims = (-2, 2.1), size = (500,500), legend = false)
#savefig("reps.pdf")
```

Out[82]:



In [83]:

```
colorac=[rand(1:3) for i in 1:8]
```

Out[83]:

```
8-element Array{Int64,1}:
 1
 2
 1
 1
 1
 3
 1
 2
```

In [84]:

```
using LinearAlgebra
```

In [86]:

```
Jclust(x,reps,colorac)
```

Out[86]:

```
2.2547374999999996
```

In [87]:

```
avg(x) = (ones(length(x)) / length(x))'*x
```

Out[87]:

```
avg (generic function with 1 method)
```

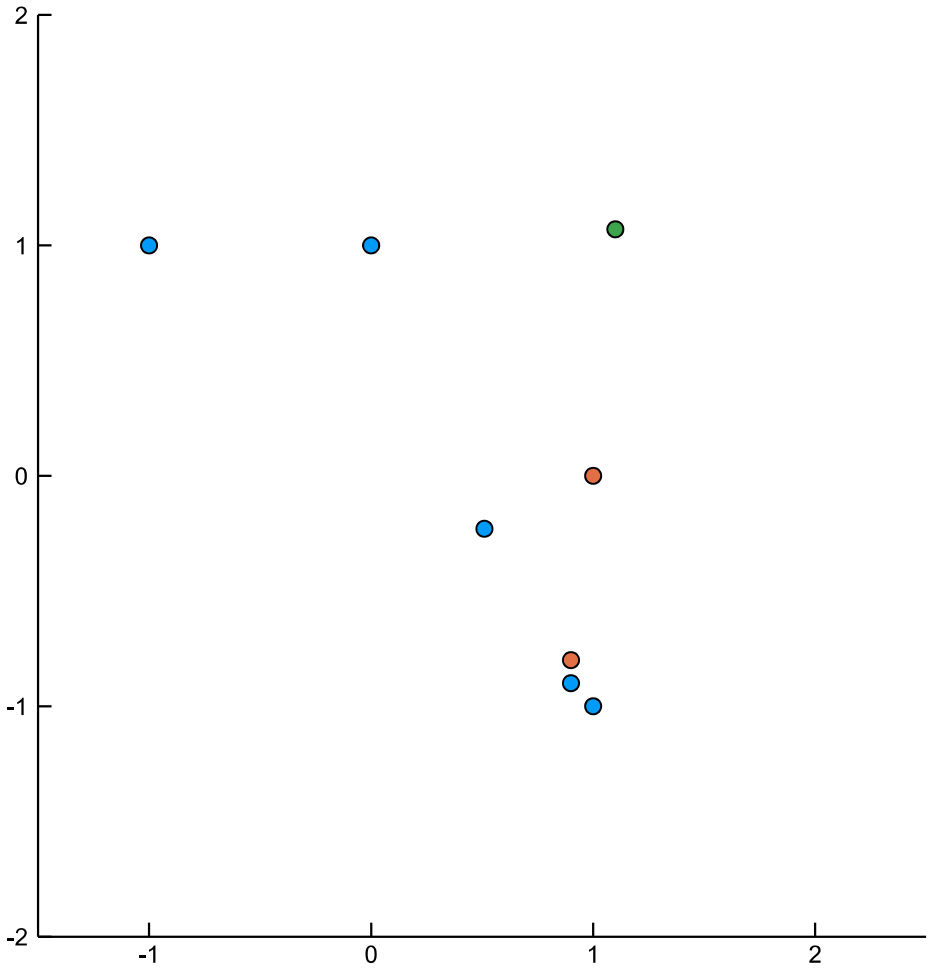
In [88]:

```
k=3
N=length(x)
for j = 1:k
 grps = [i for i=1:N if colorac[i] == j]
            end
```

In [89]:

```
k=3
N=length(x)
grps = [[x[i] for i=1:N if colorac[i] == j] for j=1:k]
scatter([c[1] for c in grps[1]], [c[2] for c in grps[1]])
scatter!([c[1] for c in grps[2]], [c[2] for c in grps[2]])
scatter!([c[1] for c in grps[3]], [c[2] for c in grps[3]])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
```

Out[89]:



In [90]:

```
function kmeans(x, k; maxiters = 100, tol = 1e-5)

 N = length(x)
 n = length(x[1])
 distances = zeros(N) # used to store the distance of each
 # point to the nearest representative.
 reps = [zeros(n) for j=1:k] # used to store representatives.

 # 'assignment' is an array of N integers between 1 and k.
 # The initial assignment is chosen randomly.
 assignment = [ rand(1:k) for i in 1:N ]

 Jprevious = Inf # used in stopping condition
 for iter = 1:maxiters

 # Cluster j representative is average of points in cluster j.
 for j = 1:k
 group = [i for i=1:N if assignment[i] == j]
 reps[j] = sum(x[group]) / length(group);
 end;

 # For each x[i], find distance to the nearest representative
                  # and its group index.
 for i = 1:N
 (distances[i], assignment[i]) =
 findmin([norm(x[i] - reps[j]) for j = 1:k])
 end;

 # Compute clustering objective.
 J = norm(distances)^2 / N

 # Show progress and terminate if J stopped decreasing.
 println("Iteration ", iter, ": Jclust = ", J, ".")
 if iter > 1 && abs(J - Jprevious) < tol * J
 return assignment, reps
 end
 Jprevious = J
 end

end
```

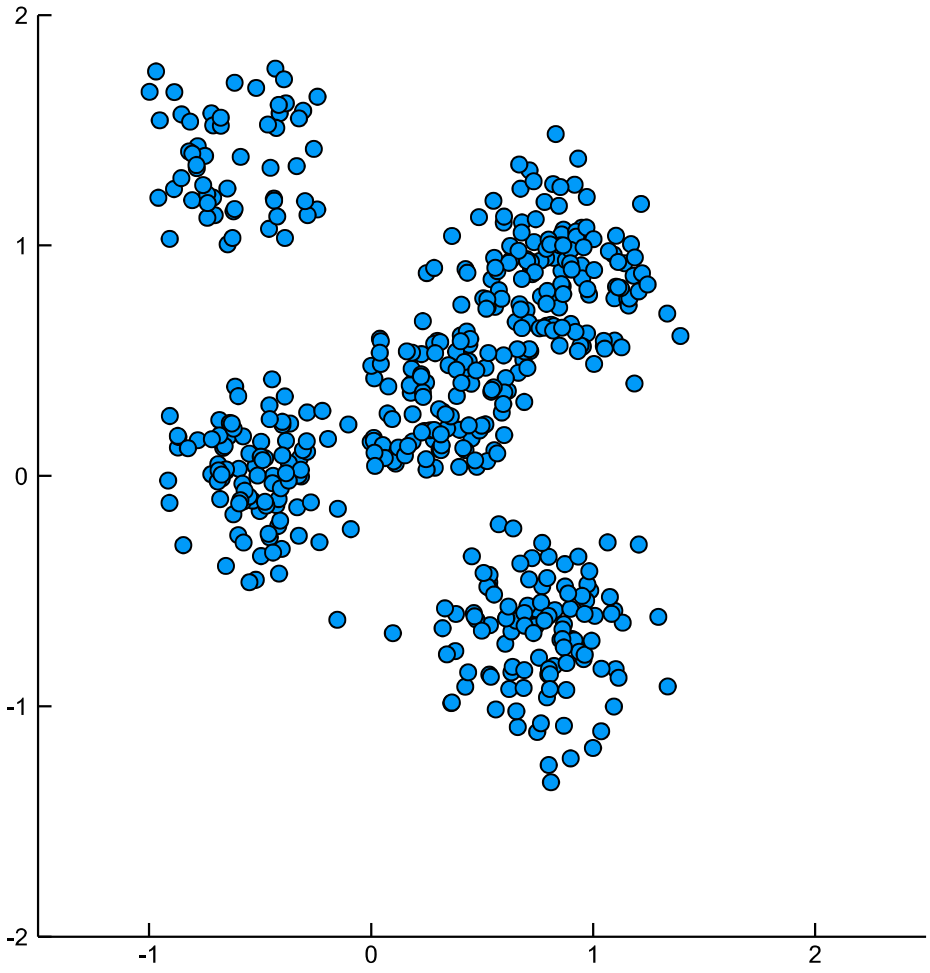
Out[90]:

```
kmeans (generic function with 1 method)
```

In [91]:

```
X = vcat( [ [-0.5,0]+0.2*randn(2) for i = 1:100 ],
[ [0.8,0.8] + 0.24*randn(2) for i = 1:150 ],
[ [0.73,-0.72] + 0.23*randn(2) for i = 1:111],
[0.6*rand(2) for i=1:89],
[[-1,1]+0.78*rand(2) for i=1:56])
scatter([x[1] for x in X], [x[2] for x in X])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
#savefig("dadosarrumados2.pdf")
```

Out[91]:



In [98]:

```
assignment, reps = kmeans(X, 5)
```

```
Iteration 1: Jclust = 0.7838671424359521.
Iteration 2: Jclust = 0.24846229038098944.
Iteration 3: Jclust = 0.2288692970094087.
Iteration 4: Jclust = 0.22396610540803835.
Iteration 5: Jclust = 0.21488548629870177.
Iteration 6: Jclust = 0.2126214408070744.
Iteration 7: Jclust = 0.21154403712051753.
Iteration 8: Jclust = 0.2113689019115699.
Iteration 9: Jclust = 0.21131914839419885.
Iteration 10: Jclust = 0.21127150575719042.
Iteration 11: Jclust = 0.21119464891337972.
Iteration 12: Jclust = 0.21102397492390995.
Iteration 13: Jclust = 0.21065727532950532.
Iteration 14: Jclust = 0.2105906462845882.
Iteration 15: Jclust = 0.21058560864391157.
Iteration 16: Jclust = 0.21058560864391157.
```

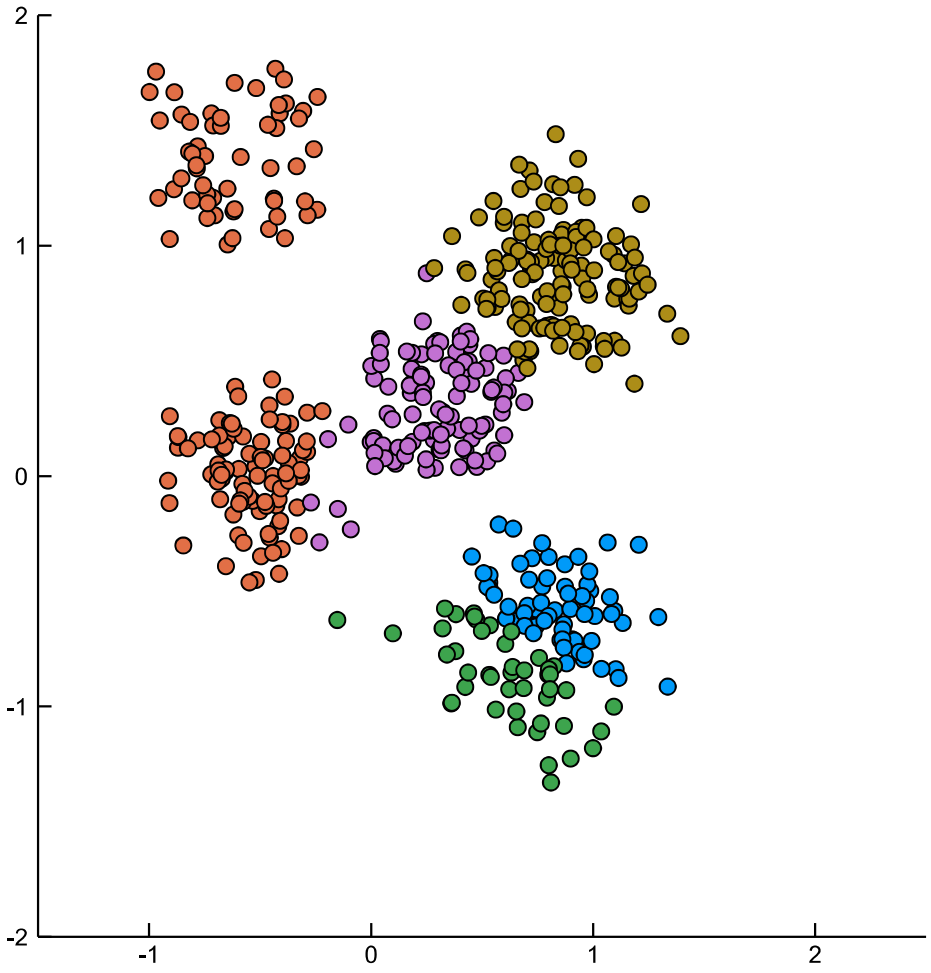
Out[98]:

```
([2, 2, 2, 2, 2, 2, 2, 2, 4, 2  …  2, 2, 2, 2, 2, 2, 2, 2, 2, 2], Array{Fl
oat64,1}[[0.841983, -0.56666], [-0.56698, 0.526705], [0.614519, -0.87913
2], [0.287391, 0.297962], [0.83546, 0.866657]])
```

In [99]:

```
k=5
N=length(X)
grps = [[X[i] for i=1:N if assignment[i] == j] for j=1:k]
scatter([c[1] for c in grps[1]], [c[2] for c in grps[1]])
scatter!([c[1] for c in grps[2]], [c[2] for c in grps[2]])
scatter!([c[1] for c in grps[3]], [c[2] for c in grps[3]])
scatter!([c[1] for c in grps[4]], [c[2] for c in grps[4]])
scatter!([c[1] for c in grps[5]], [c[2] for c in grps[5]])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
#savefig("funcaodecolarac.pdf")
```

Out[99]:



In [102]:

```
assignment, reps = kmeans(x, 3)
```

```
Iteration 1: Jclust = 0.6337729166666668.
Iteration 2: Jclust = 0.33092135416666674.
Iteration 3: Jclust = 0.2978875.
Iteration 4: Jclust = 0.2978875.
```

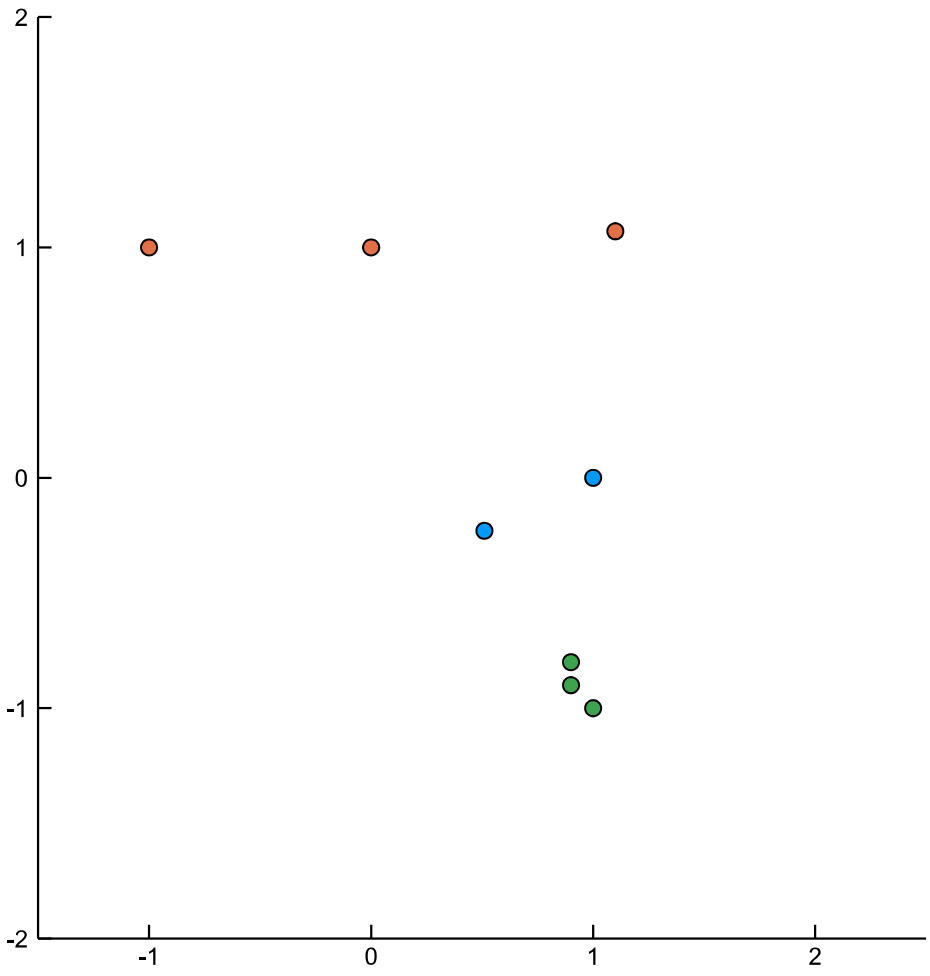
Out[102]:

```
([2, 1, 2, 3, 3, 2, 1, 3], Array{Float64,1}[[0.755, -0.115], [0.0333333,
1.02333], [0.933333, -0.9]])
```

In [103]:

```
k=3
N=length(x)
grps = [[x[i] for i=1:N if assignment[i] == j] for j=1:k]
scatter([c[1] for c in grps[1]], [c[2] for c in grps[1]])
scatter!([c[1] for c in grps[2]], [c[2] for c in grps[2]])
scatter!([c[1] for c in grps[3]], [c[2] for c in grps[3]])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
                # savefig("aglutbom.pdf")
```

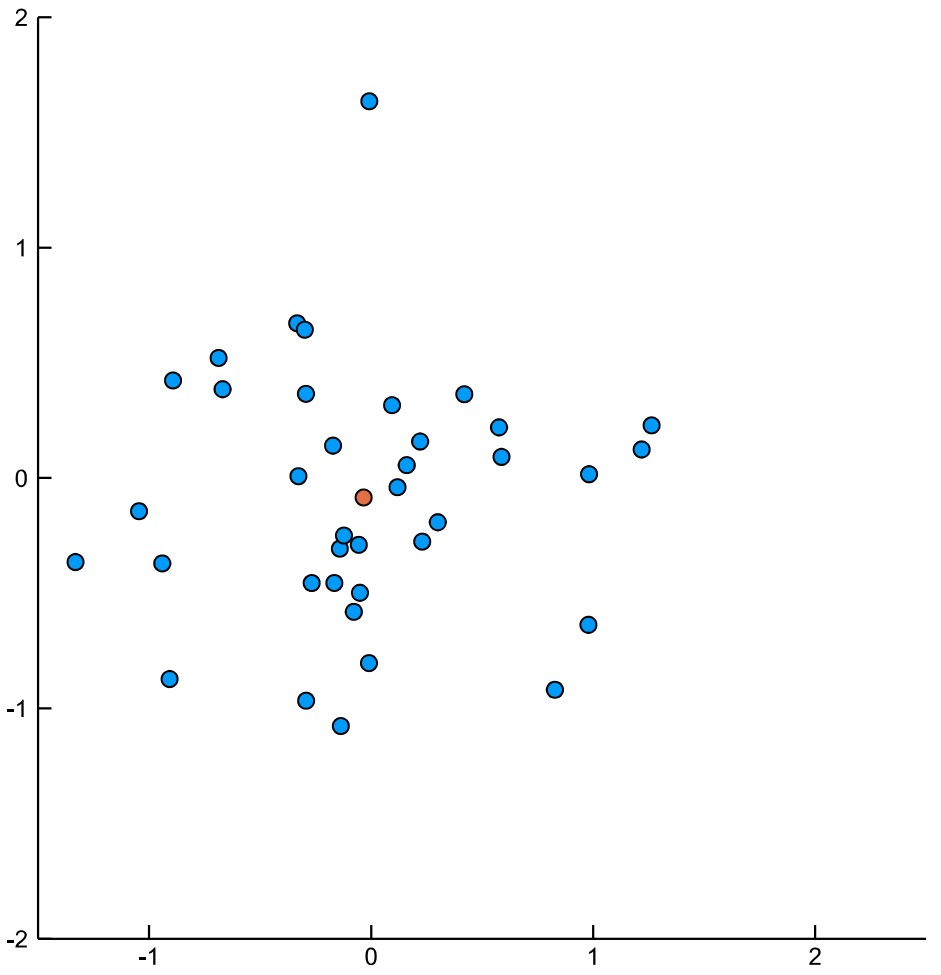
Out[103]:



In [104]:

```
Y = vcat( [ 0.7*randn(2) for i = 1:37 ])
y1=[y[1] for y in Y]
y2=[y[2] for y in Y]
z1=(1/37)*sum(y1)
z2=(1/37)*sum(y2)
scatter([y[1] for y in Y], [y[2] for y in Y])
scatter!([z1],[z2])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
#savefig("centroide.pdf")
```

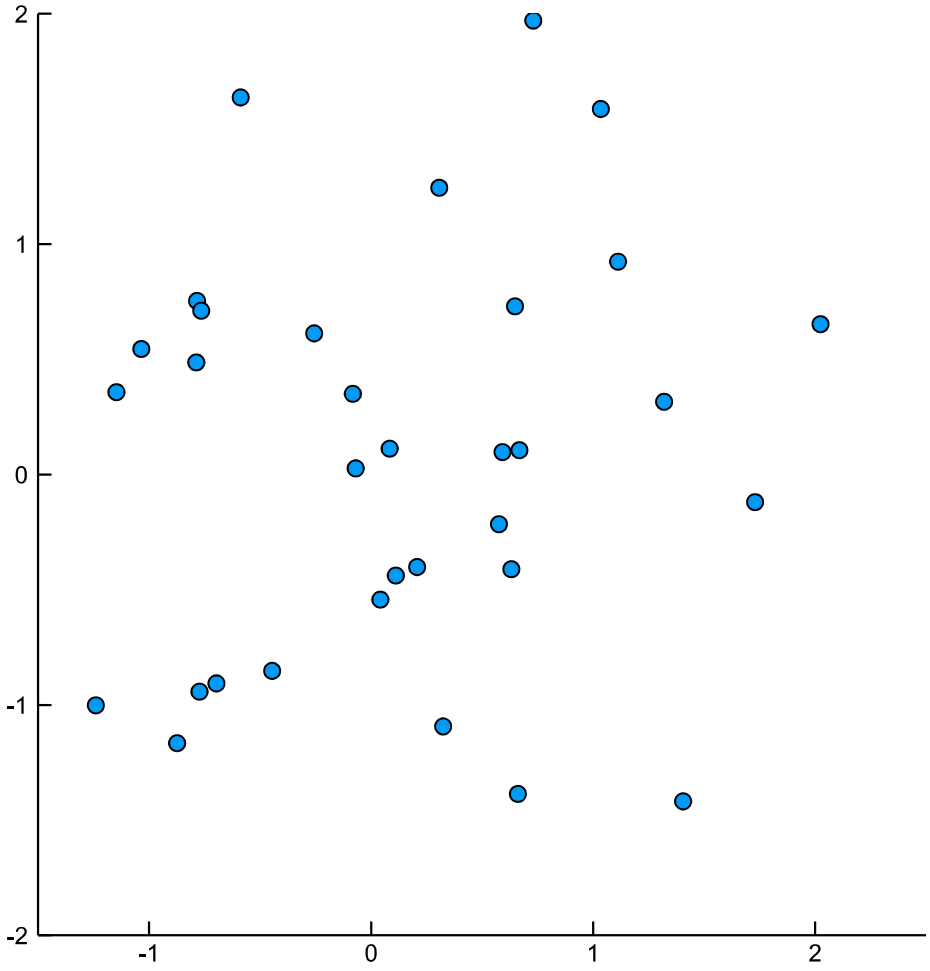
Out[104]:



In [105]:

```
Y = vcat( [ randn(2) for i = 1:37 ])
scatter([y[1] for y in Y], [y[2] for y in Y])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
```

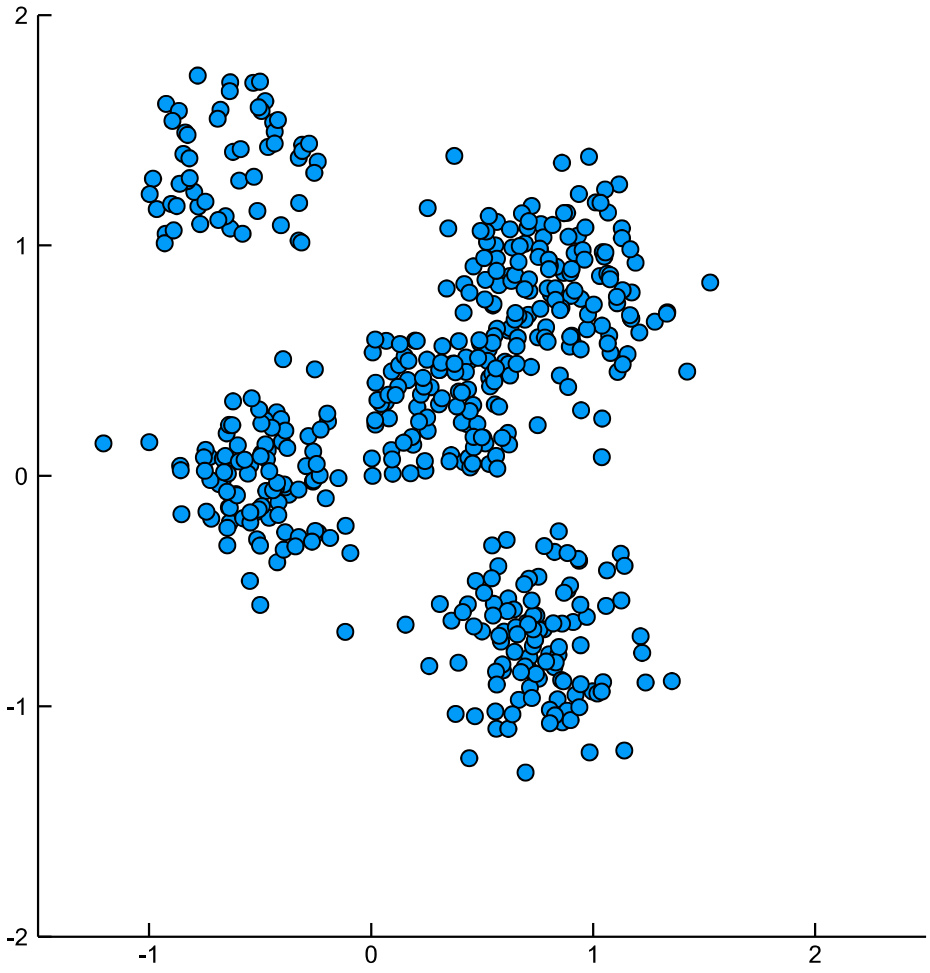
Out[105]:



In [106]:

```
X = vcat( [ [-0.5,0]+0.2*randn(2) for i = 1:100 ],
[ [0.8,0.8] + 0.24*randn(2) for i = 1:150 ],
[ [0.73,-0.72] + 0.23*randn(2) for i = 1:111],
[0.6*rand(2) for i=1:89],
[[-1,1]+0.78*rand(2) for i=1:56])
scatter([x[1] for x in X], [x[2] for x in X])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
#savefig("dadosarrumados2.pdf")
```

Out[106]:



In [107]:

```
n=length(X)
v=10*rand(n)
u=zeros(n)
for i=1:n
    u[i]=trunc(Int,v[i])%5
end
u
```

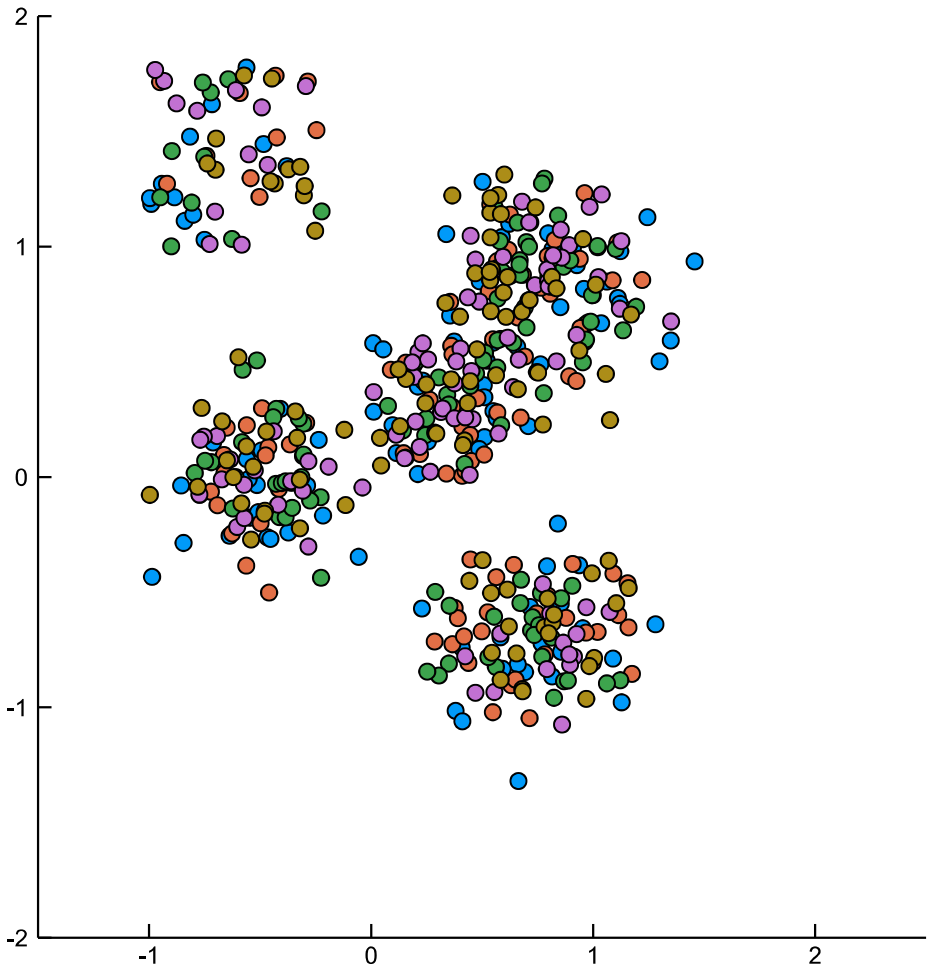
Out[107]:

```
506-element Array{Float64,1}:
 2.0
 4.0
 4.0
 0.0
 1.0
 2.0
 2.0
 3.0
 4.0
 3.0
 4.0
 4.0
 4.0
 ⋮
 4.0
 2.0
 4.0
 0.0
 3.0
 4.0
 0.0
 0.0
 4.0
 1.0
 3.0
 4.0
```

In [112]:

```
n=length(X)
grps = [[X[i] for i=1:n if u[i] == j] for j=0:4]
scatter([c[1] for c in grps[1]], [c[2] for c in grps[1]])
scatter!([c[1] for c in grps[2]], [c[2] for c in grps[2]])
scatter!([c[1] for c in grps[3]], [c[2] for c in grps[3]])
scatter!([c[1] for c in grps[4]], [c[2] for c in grps[4]])
scatter!([c[1] for c in grps[5]], [c[2] for c in grps[5]])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
#savefig("randomcolors.pdf")
```

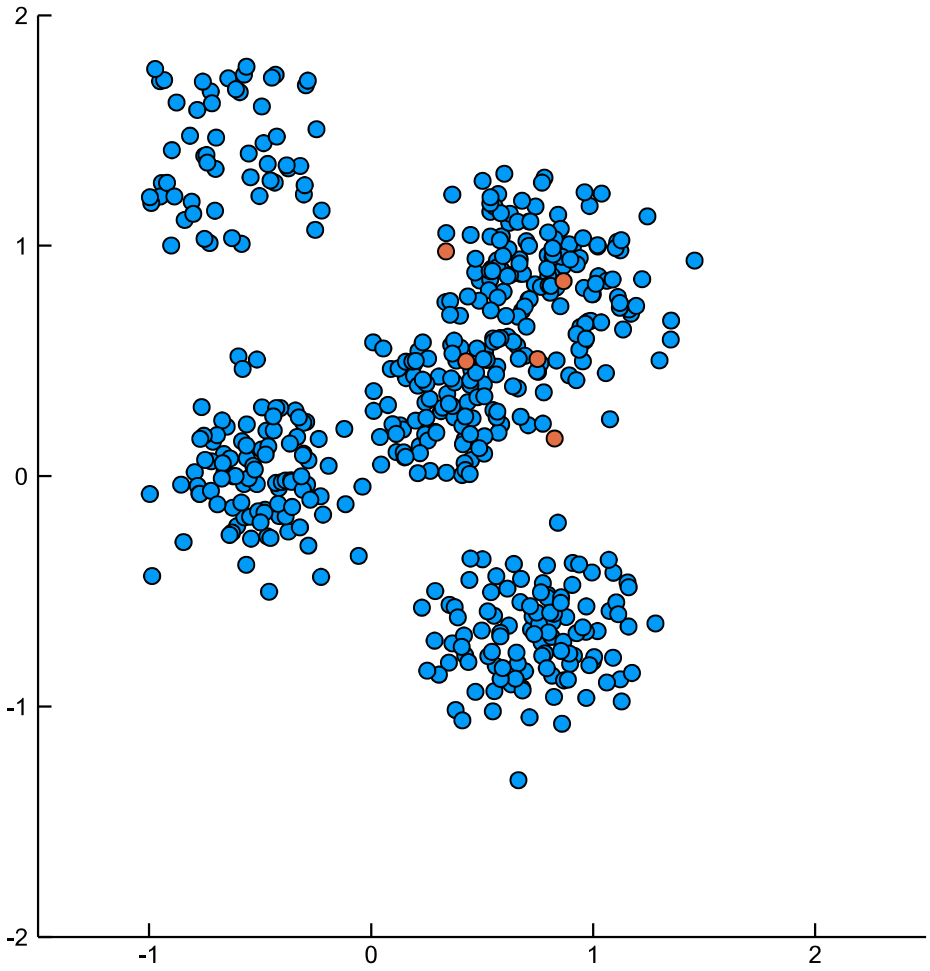
Out[112]:



In [111]:

```
X = vcat( [ [-0.5,0]+0.2*randn(2) for i = 1:100 ],
[ [0.8,0.8] + 0.24*randn(2) for i = 1:150 ],
[ [0.73,-0.72] + 0.23*randn(2) for i = 1:111],
[0.6*rand(2) for i=1:89],
[[-1,1]+0.78*rand(2) for i=1:56])
Y=vcat([rand(2) for i=1:5])
scatter([x[1] for x in X], [x[2] for x in X])
scatter!([x[1] for x in Y], [x[2] for x in Y])
plot!(legend = false, grid = false, size = (500,500),
xlims = (-1.5,2.5), ylims = (-2,2))
#savefig("representantes.pdf")
```

Out[111]:



In [ ]: