Hasher-Lind Normal

Ryan McClarren

November 3, 2016

The function that we will use as our test of Hasofer-Lind is

$$Z = x_1 - x_2 + 0.1\cos(4x_1)$$

. Let's say that

$$X_1 \sim N(-0.2, 0.5)$$

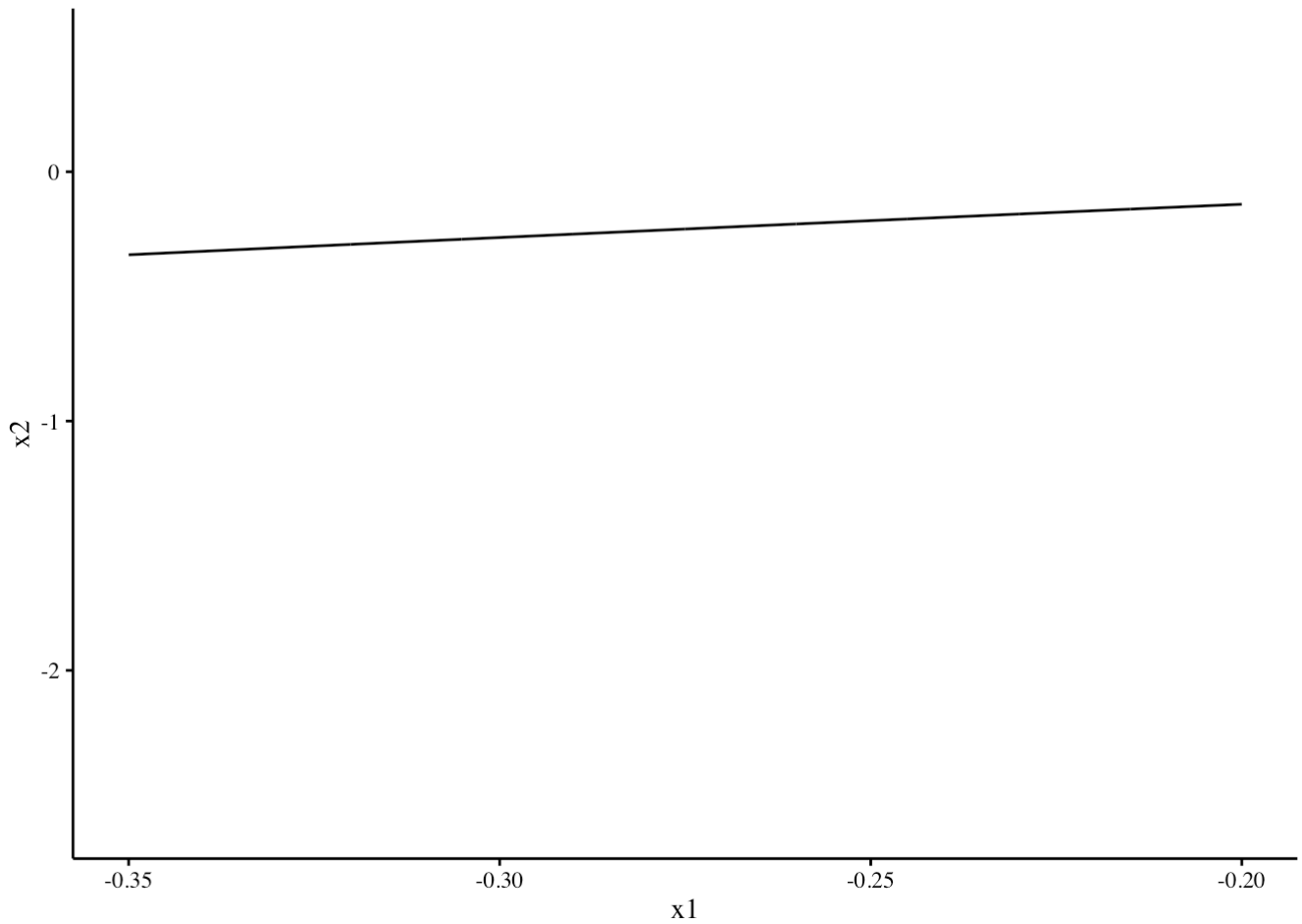
and

$$X_2 \sim N(-2.5, 2.5).$$

The failure surface for this function (i.e., where $Z = 0$) occurs at

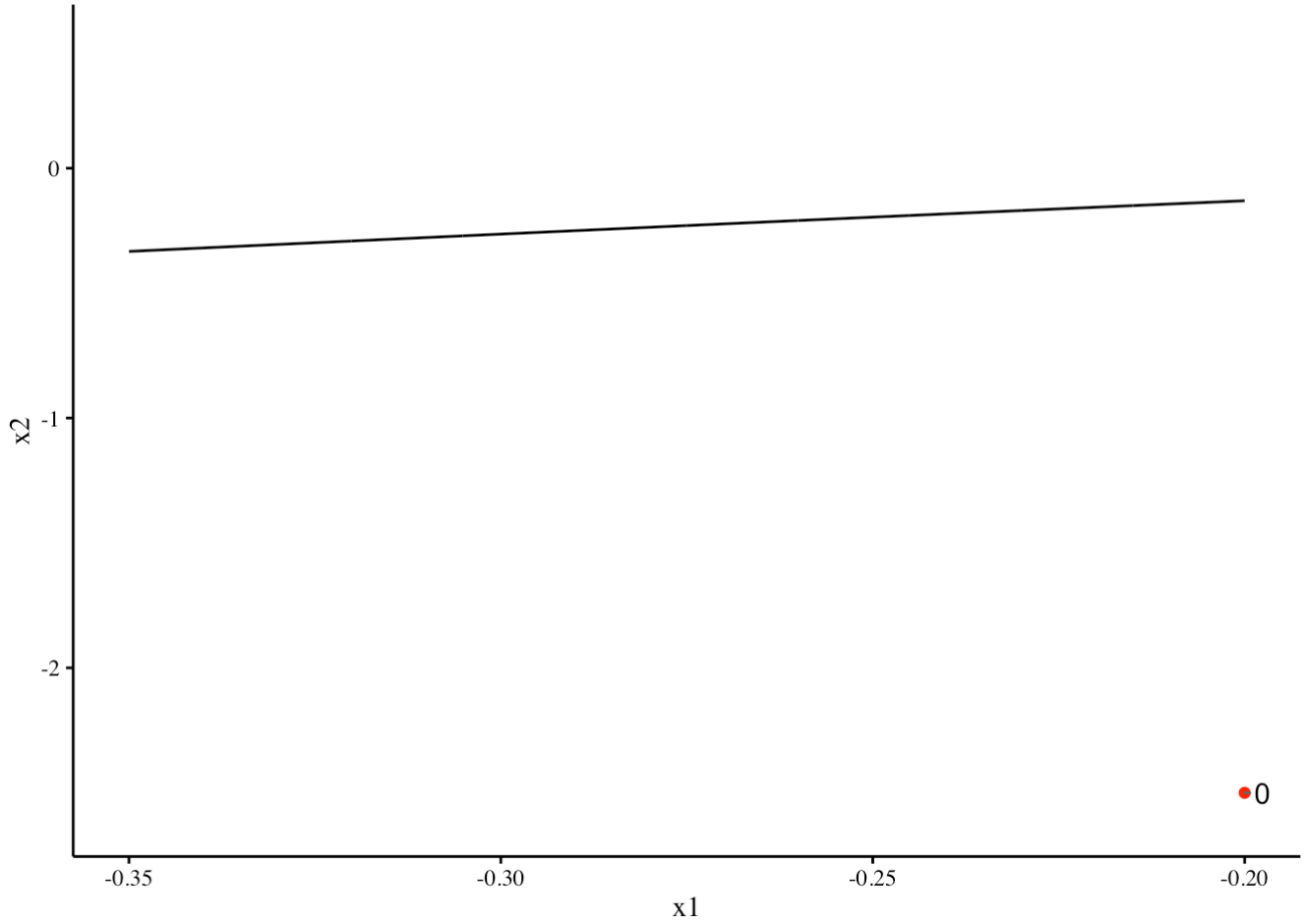
$$x_2 = x_1 + 0.1\cos(4x_1)$$

.



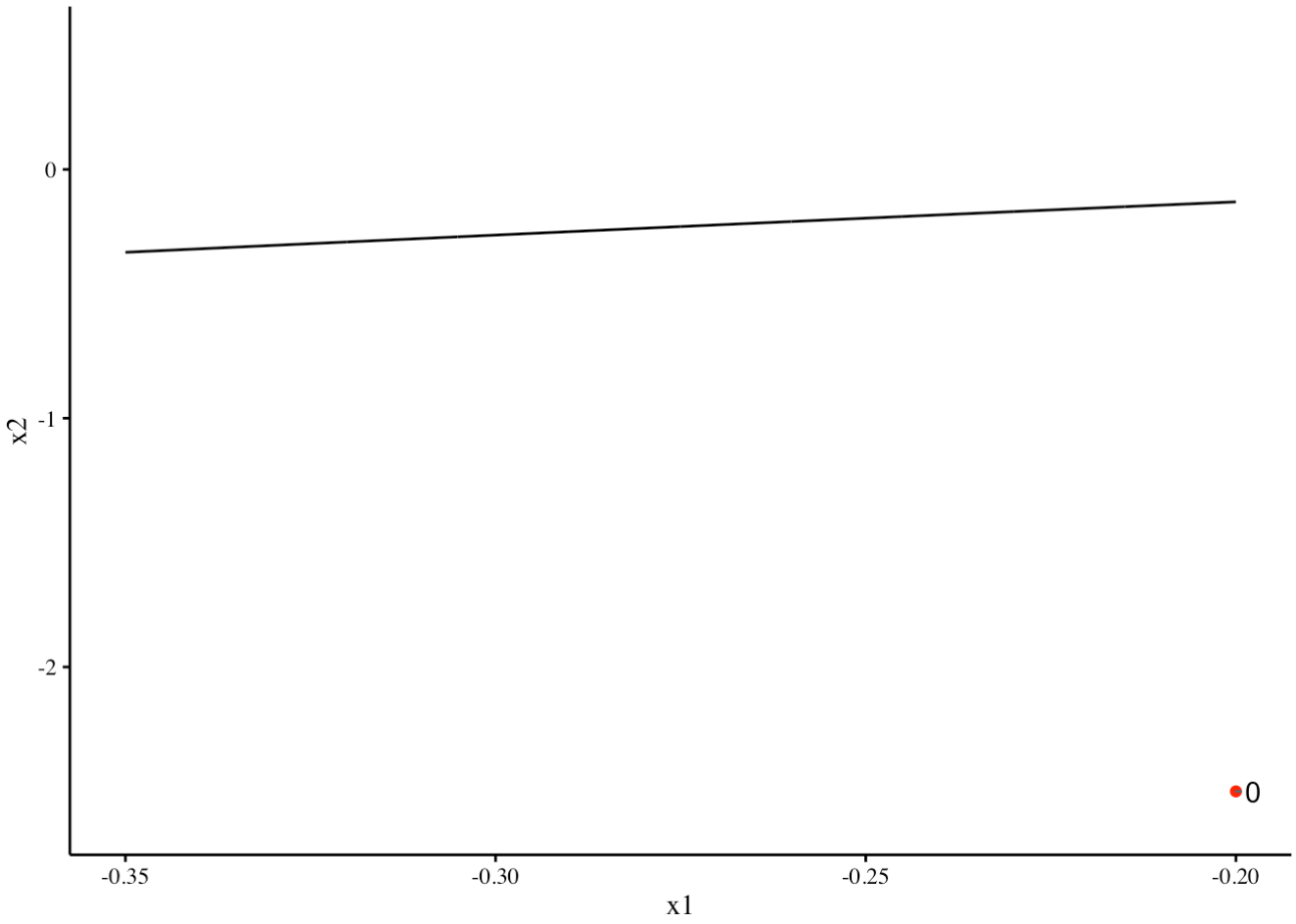
The HL procedure begins at the mean point.

```
mux1 = -0.2
mux2 = -2.5
mainplot = mainplot + geom_point(data=data.frame(X1=mux1,X2=mux2), color="red") +
  geom_text_repel(data=data.frame(X1=mux1,X2=mux2), label=0)
print(mainplot)
```



Next we normalize the x_i and find a search direction.

```
sigmax1 = 0.5
sigmax2 = 2.5
iteration1 = data.frame(X1 = mux1, X2 = mux2)
iteration1$X1prime = (iteration1$X1-mux1)/sigmax1
iteration1$X2prime = (iteration1$X2-mux2)/sigmax2

dg = c(dZdx1(iteration1), dZdx2(iteration1))
dgprime = dg*c(sigmax1, sigmax2)
absgsquared = sum(dgprime*dgprime)
xprime = 1/absgsquared*(sum(dgprime*c(iteration1$X1prime,iteration1$X2prime))
                        - Z(iteration1))*dgprime
iteration2 = data.frame(X1 = xprime[1]*sigmax1+mux1,X2 = xprime[2]*sigmax2+mux2 )

beta = sqrt(sum(xprime^2))
absg = abs(Z(iteration2))

#mainplot = mainplot + geom_point(data=iteration2, color="red") + geom_text_repel(data=iteration
2, label=1)
print(mainplot)
```




```
iteration2$lab = "1"
allits = iteration2
```

After one iteration $\beta = 0.9179493$, and $|Z| = 0.0060349$.

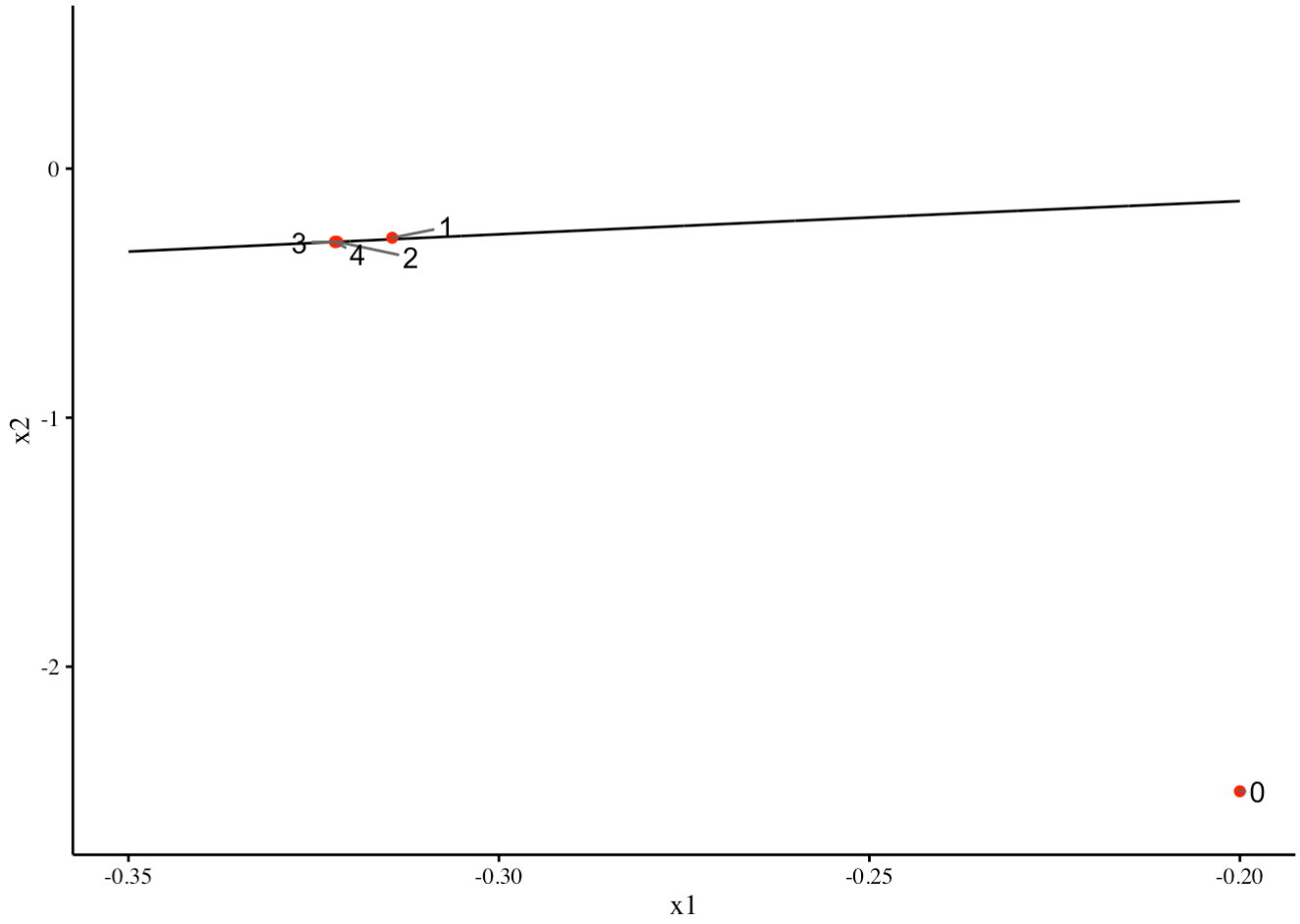
Now let's turn this into an procedure that runs until either $\Delta\beta$ or $|Z|$ is less than 10^{-6}.

```
delta = 1e-6
converged = 0
it = 2
beta_old = beta
old_iteration = iteration2
while (!(converged)){
old_iteration$X1prime = (old_iteration$X1-mux1)/sigmax1
old_iteration$X2prime = (old_iteration$X2-mux2)/sigmax2

dg = c(dZdx1(old_iteration), dZdx2(old_iteration))
dgprime = dg*c(sigmax1, sigmax2)
absgsquared = sum(dgprime*dgprime)
xprime = 1/absgsquared*(sum(dgprime*c(old_iteration$X1prime,old_iteration$X2prime))
                        - Z(old_iteration))*dgprime
new_iteration = data.frame(X1 = xprime[1]*sigmax1+mux1,X2 = xprime[2]*sigmax2+mux2 )

beta = sqrt(sum(xprime^2))
absg = abs(Z(new_iteration))

print(paste("Iteration:",it,"beta = ",beta,"|Z| = ",absg))

it = it + 1

if ( (abs(beta_old-beta) < delta) & (absg < delta))
  converged = 1
if (it > 10)
  converged = 1
beta_old = beta
old_iteration = new_iteration
new_iteration$lab = it-1
allits = rbind(allits,new_iteration)
}
```

```
## [1] "Iteration: 2 beta =  0.915482236146609 |Z| =  1.31604513333196e-05"
## [1] "Iteration: 3 beta =  0.915476970253062 |Z| =  1.82026514833344e-08"
## [1] "Iteration: 4 beta =  0.915476962979242 |Z| =  2.43176243031051e-11"
```

The iteration procedure looks like this on a graph.

```
mainplot = mainplot + geom_point(data=allits, color="red") + geom_text_repel(data=allits, aes(la
bel=lab))
print(mainplot)
```



The probability of failure is $1 - \Phi(\beta) = 0.1799706$. Let's check that with Monte Carlo.

```
N = 1e6
df <- data.frame(X1 = rnorm(N,mux1, sigmax1),X2 = rnorm(N,mux2, sigmax2) )
df$Z = Z(df)
pfail = length(df$Z[df$Z<0])/N
```

The actual probability of failure is 0.182784. The Hasofer-Lind approximation is off by 1.5391689%.