DefaultRate と重点サンプリング

Naoya Hieda

2017 年 05 月 25 日

目的

- 重点サンプリングが、デフォルト率の密度関数の推定においてどれだけ有能か確認する。
- デフォルト率の推定における重点関数 (提案分布) の選択について考察する
- 同様に、サンプル数 (Particle 数) についても考察する

方法

Joho.C.Hull の本で定義されているデフォルト率の式に関して
重点サンプリングでいろいろを行う

デフォルト率の従う式

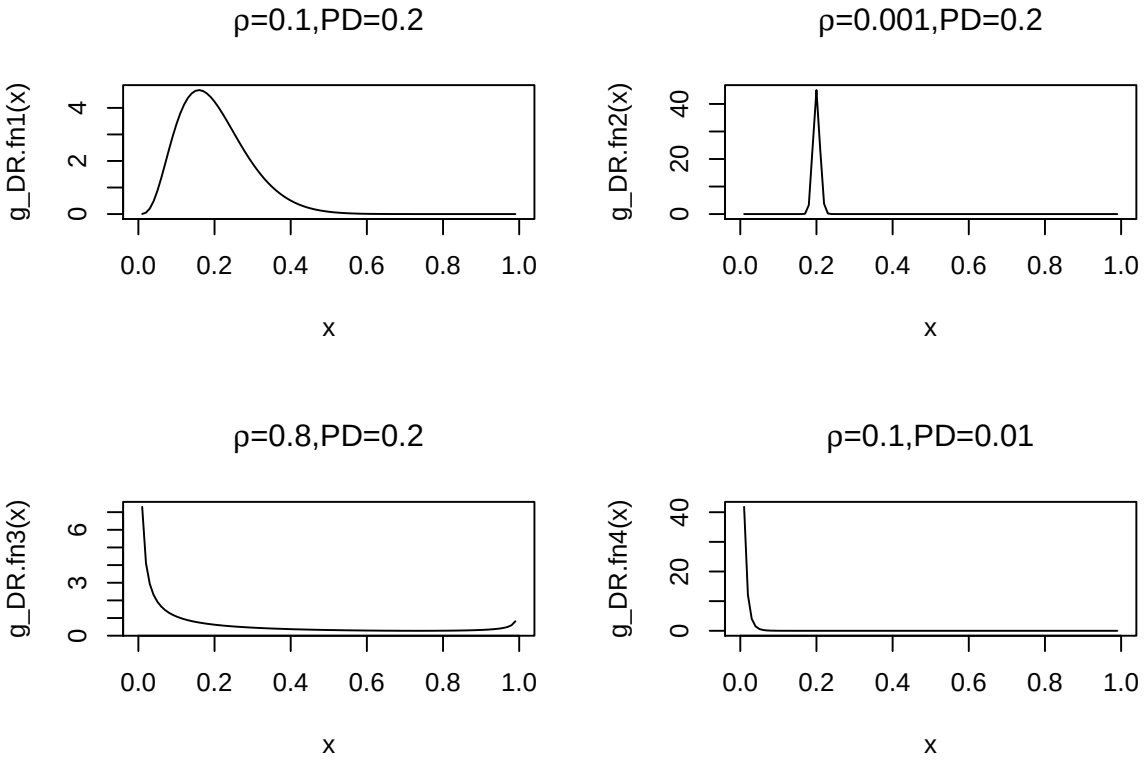
$$\mathrm{g}(\mathrm{DR}|\mathrm{PD},\rho) = \sqrt{\frac{1-\rho}{\rho}}\exp\left\{\frac{1}{2}\left[(N^{-1}(DR))^2 - \left(\frac{\sqrt{1-\rho}N^{-1}(DR) - N^{-1}(PD)}{\sqrt{\rho}}\right)^2\right]\right\}$$

対象とするデフォルト密度下記の 4 パターンのデフォルト率の密度関数について重点サンプリングで色々行う

```
g_DR.fn1 <- function(DR)g_DR.fn(0.1,0.2,DR)
g_DR.fn2 <- function(DR)g_DR.fn(0.001,0.2,DR)
g_DR.fn3 <- function(DR)g_DR.fn(0.8,0.2,DR)
g_DR.fn4 <- function(DR)g_DR.fn(0.1,0.01,DR)

title_name <- c(expression(paste(rho,"=0.1,",PD,"=0.2")),expression(paste(rho,"=0.001,",PD,"=0.2")),
expression(paste(rho,"=0.8,",PD,"=0.2")),expression(paste(rho,"=0.1,",PD,"=0.01")))
pds <- c(0.2,0.2,0.2,0.01)
rhos <- c(0.1,0.001,0.8,0.1)

par(mfrow=c(2,2))
curve(g_DR.fn1,main=expression(paste(rho,"=0.1,",PD,"=0.2")))
curve(g_DR.fn2,main=expression(paste(rho,"=0.001,",PD,"=0.2")))
curve(g_DR.fn3,main=expression(paste(rho,"=0.8,",PD,"=0.2")))
curve(g_DR.fn4,main=expression(paste(rho,"=0.1,",PD,"=0.01")))
```



重点サンプリング

元々のアイデアとしては・・・
通常のモンテカルロ法で、元の分布からサンプリングしようとすると、分布の端がサンプリングされるのに時間がかかり、推定量が収束するまでにかかるサンプル数がたくさん必要

重点サンプリングは、この問題を解決するため欲しい範囲のサンプルを重点的にサンプリングできる重点関数というものを導入しましょうというアイデア。

- 推定量の分散を通常のモンテカルロ法によるものより改善できる

重点サンプリング基本恒等式

$$E_f[h(X)] = \int_\chi h(x)\frac{f(x)}{g(x)}g(x)dx = E_g[\frac{h(X)f(X)}{g(X)}]$$

から、次の推定量を求める。これが収束するのは通常のモンテカルロ法が収束するのと同じ理由 (大数の強法則)

$$\frac{1}{n}\sum_{j=1}^{m}\frac{f(X_j)}{g(X_j)}h(X_j) \to E_f[h(X)]$$

とりあえずは一様分布 (0,1) を重点関数 g、サンプルサイズは 1000 とする
h は DR として、DR の期待値を求める。John.C.Hull によると、この期待値は各密度関数のパラメータ PD と一致するはず
Pattern3 以外はそれっぽい
Pattern3 だけばらつく理由と、その解決策は後ほど

```
set.seed(10000)
N <- 1000
for(t in 1:5){
sim <- runif(N,0,1)
weight_1 <-  g_DR.fn1(sim) / dunif(sim)
weight_2 <-  g_DR.fn2(sim) / dunif(sim)
weight_3 <-  g_DR.fn3(sim) / dunif(sim)
weight_4 <-  g_DR.fn4(sim) / dunif(sim)
weight <- data.frame(weight_1/sum(weight_1),weight_2/sum(weight_2),
                     weight_3/sum(weight_3),weight_4/sum(weight_4))
colnames(weight)<- c('pattern1','pattern2','pattern3','pattern4')
print(colSums(weight * sim))
}
```

```
##    pattern1    pattern2    pattern3    pattern4
## 0.199687542 0.201409288 0.226341239 0.009643512
##    pattern1    pattern2    pattern3    pattern4
## 0.194197738 0.199215169 0.264241752 0.009701833
##   pattern1   pattern2   pattern3   pattern4
## 0.20184624 0.19980659 0.23013647 0.01239423
##   pattern1   pattern2   pattern3   pattern4
## 0.20203629 0.19871167 0.29449808 0.01139801
##    pattern1    pattern2    pattern3    pattern4
## 0.205111633 0.199716495 0.104375504 0.008375601
```

重点サンプリングによって、一定の値以上をとる確率を計算することができるが、これを 95% 点の算出に使えないだろうか？
とりあえずは、ちょっと難しい。

それぞれの収束の様子
やはり、一番左上の、概形が綺麗な密度関数は、収束しやすい
極端なパラメータをとって、密度関数がいびつな (極端な?) 形をとっていると、収束の様子に違和感がある。

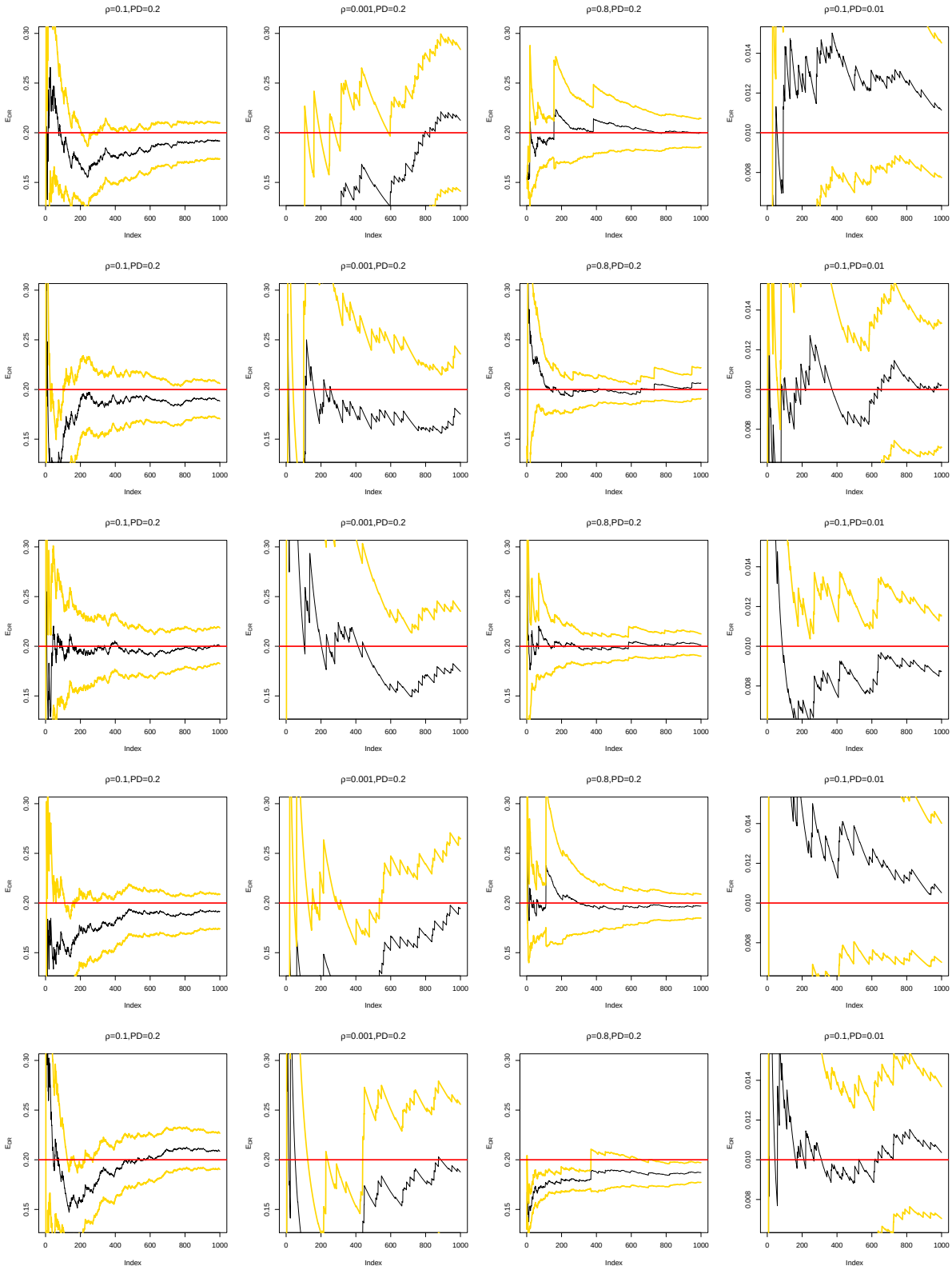
- パターン 2 の ρ が極端に低い場合は、(0,1) の一様関数だと、確率密度が 0 に近い値が出やすいため、0.2 付近の値をとった時だけ大きな値を取るため、その時だけ大きな変動を起こして安定しにくい。

- パターン 3 の ρ が極端に高い値をとる場合は、基本的には安定しているが、密度関数の概形から分かるように、1 付近に密度を持つため、サンプリング数が足りたいと、1 付近がサンプリングされた時に、極端な変動を起こす (200,400 付近)

- パターン 4 の PD が極端に低い値をとる場合は、比較的安定しているが、左上に比べると収束に時間がかかっていることが分かる。(元の値が小さく安定しているので、他よりも y 軸の範囲を狭めている)

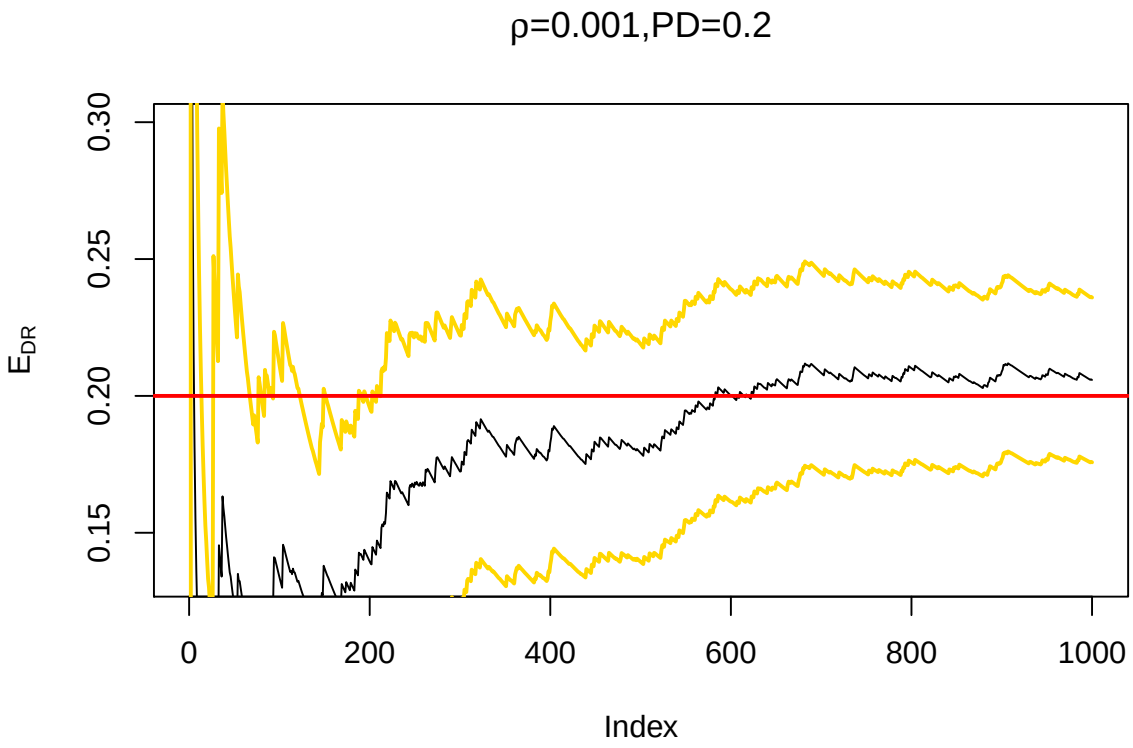
```
par(mfrow=c(5,4))
set.seed(10000)
for(t in 1:5){
```

```
sim <- runif(N,0,1)
weight_1 <-  g_DR.fn1(sim) / dunif(sim)
weight_2 <-  g_DR.fn2(sim) / dunif(sim)
weight_3 <-  g_DR.fn3(sim) / dunif(sim)
weight_4 <-  g_DR.fn4(sim) / dunif(sim)
weight <- data.frame(weight_1,weight_2,weight_3,weight_4)
for(i in 1:4){
  estint <- cumsum(weight[,i]*sim)/1:N
  esterr <- sqrt(cumsum((weight[,i]*sim-estint)^2))/(1:N)
  plot(cumsum(weight[,i]*sim)/1:N,type="l",main = title_name[i],ylab = expression(E[DR]),ylim = c(pds[i]/1.5,pds[i]*1.5))
  lines(estint+2*esterr, col="gold", lwd=2)
  lines(estint-2*esterr, col="gold", lwd=2)
  abline(h=pds[i],col='red',lwd=2)
}
}
```



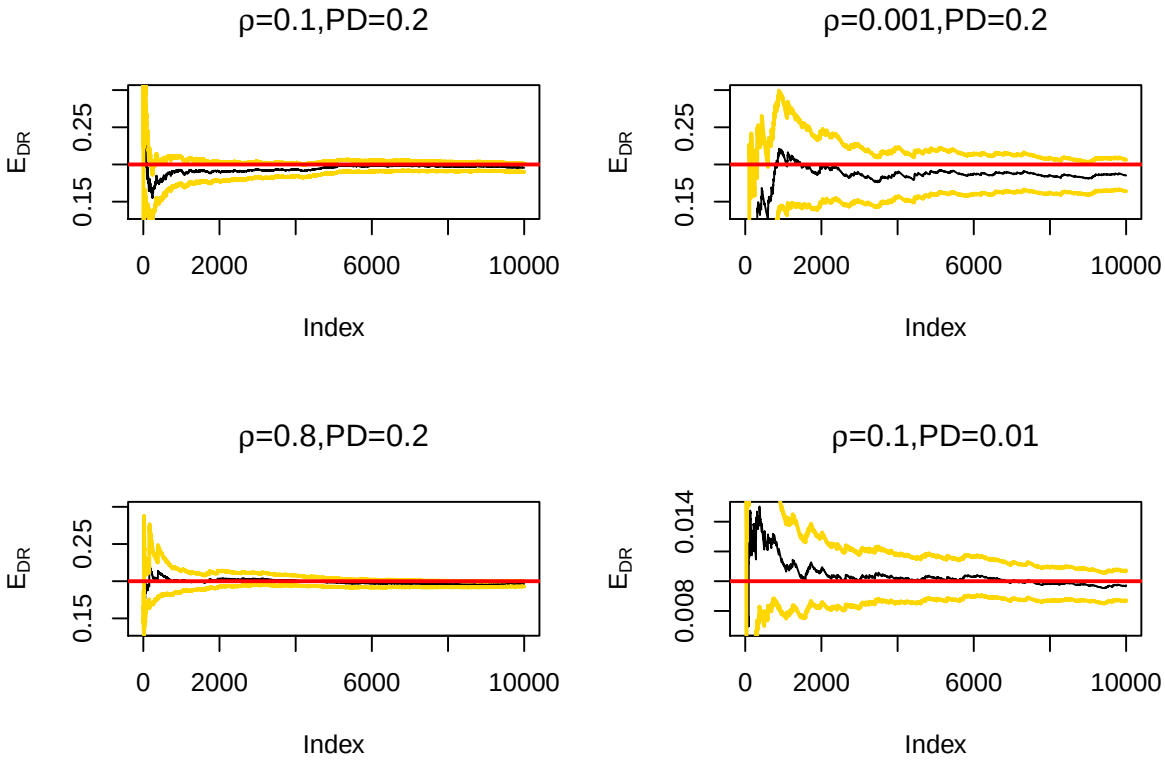
パターン 2 の場合において、重点関数の範囲が狭まれば、うまくいくはず
仮に一様分布 (0.15,0.25) にすると、収束する

```
## [1] 0.2058494
```



まぁサンプルサイズをふやせば収束する

```
## weight_1.sum.weight_1. weight_2.sum.weight_2. weight_3.sum.weight_3.
##             0.20024174             0.19987535             0.21989643
## weight_4.sum.weight_4.
##             0.01017655
```

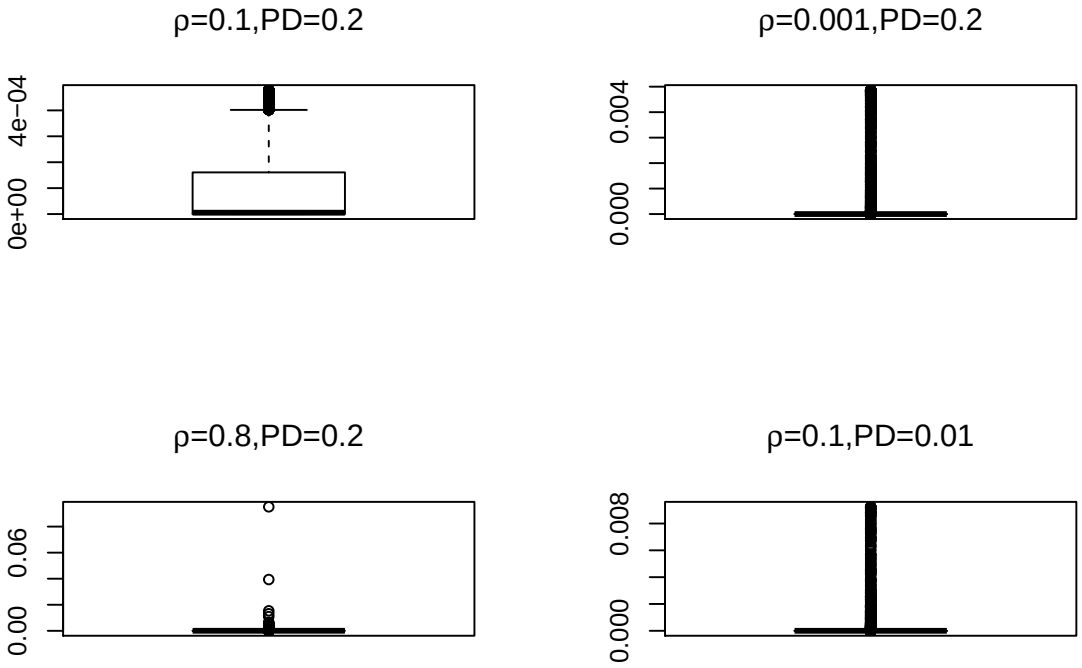


サンプリング重点サンプリング

有効サンプルサイズなんかを確かめているがこの辺解説が 4 章なので現状は保留。
weight だけ boxplot で確認。

サンプル数 10000 での結果なので、右下とか良くない。

```
par(mfrow=c(2,2))
for(i in 1:4){
  boxplot(weight[,i]/sum(weight[,i]),main = title_name[i])
}
```



重点関数の選択

理論的に最適なものは求まるが実践での有用性は低い
去年も式とだけにらめっこして、実装の仕方がわからず断念した

g の妥当性を判断するには、結果の推定量の分散を検討する
もとの分布が期待値を持っていれば、ほぼ確実に収束するが、
この推定量の分散は、以下の期待値が有限の場合にだけ有限になる。

$$E_g\left[h^2(X)\frac{f^2(X)}{g^2(X)}\right] = E_f\left[h^2(X)\frac{f(X)}{g(X)}\right] = \int_\chi h^2(x)\frac{f^2(x)}{g(x)}dx < \infty$$

今は、$f(x)/g(x) = f(x)$

という事は、一様分布を仮定する限り、この問題とは無関係。ただし、ρ が限りなく 0 に近いときはあやしい