

希少サンプルを分割する

ryamada

2016年11月12日

希少サンプル

DNA鑑定試料がある。試料の量が少ない。1回の実験に全部使うのがよいのか、再実験用に取り分けておくのがよいのか？

基礎編

どうして再実験用に取り分けておきたいと感 じるのか？

1回目で失敗しそう...

2回目は失敗しないのか？

失敗を想定して再実験する、とは

何回かに分けて実験する。

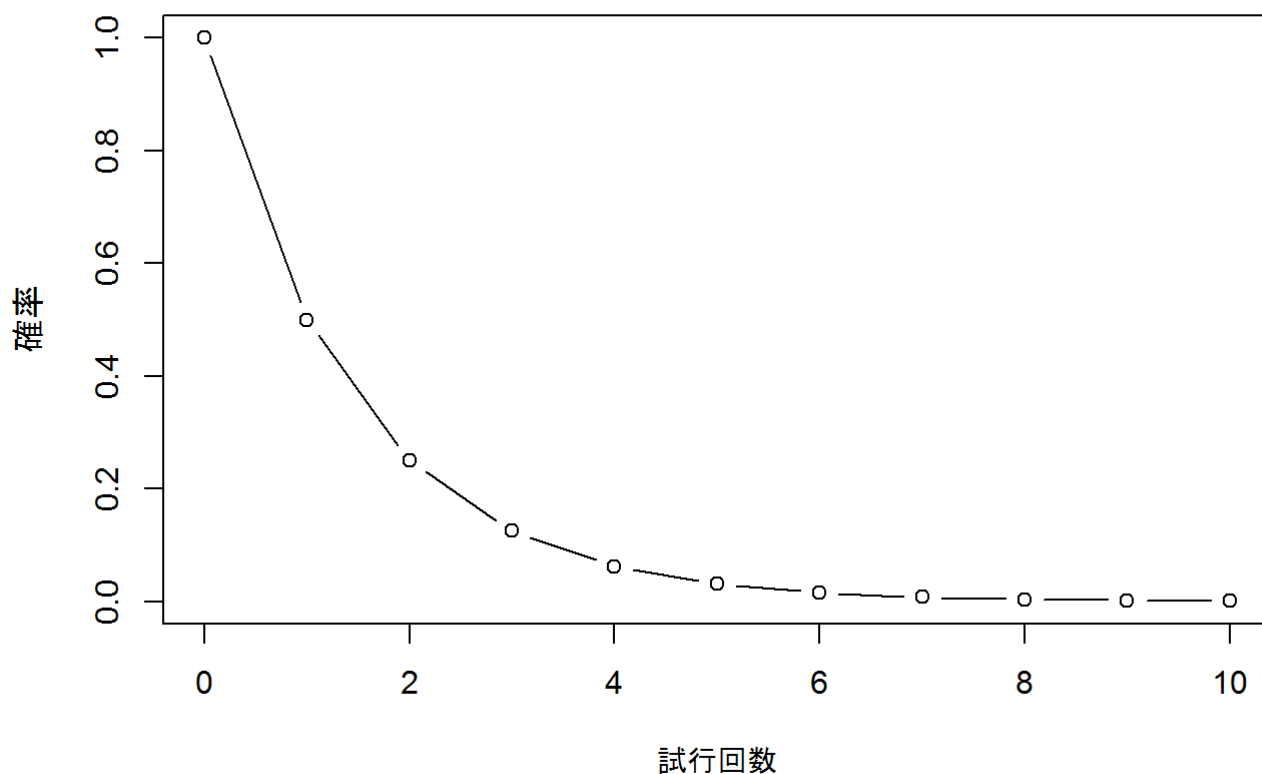
そのうちの1回でも成功すれば(成功して正しい結果が得られれば)、よし、とすることにする。

毎回、同じ確率で失敗するなら：

```
p.failure <- 0.5
```

```
plot(0:10, p.failure^(0:10), type="b", main="すべて失敗する確率", xlab="試行回数", ylab="確率")
```

すべて失敗する確率



複数回やれば、全部失敗する確率は、どんどん小さくなる。

$$p^k$$

どうして分割をためらうのか？

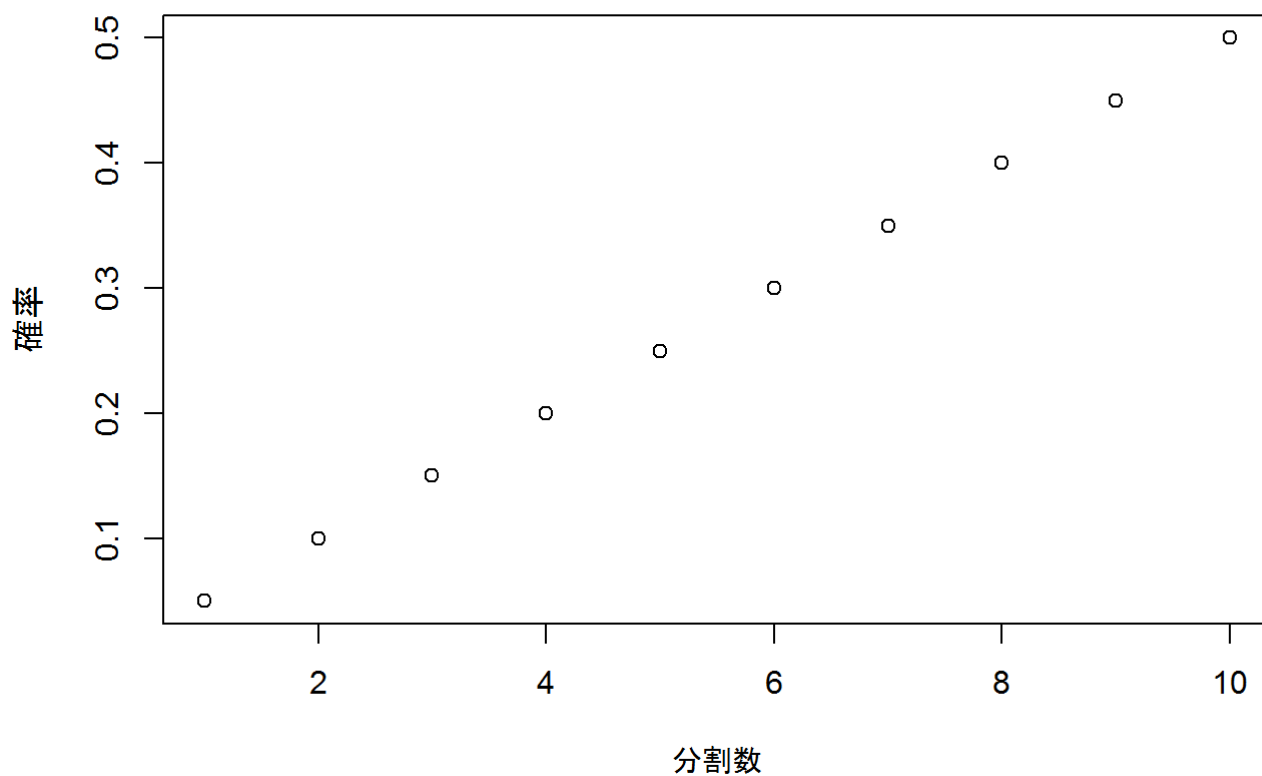
全量を1回に使っても、ちゃんとした結果が出ないかもしれないのに、分割したら、全滅するかもしれない。それよりは、できるだけ好条件の実験を1回だけやる方がよいのでは？

失敗の確率が1回の実験量と、たとえば次のような関係があるとする

$$p. failure \times volume. exp = C$$

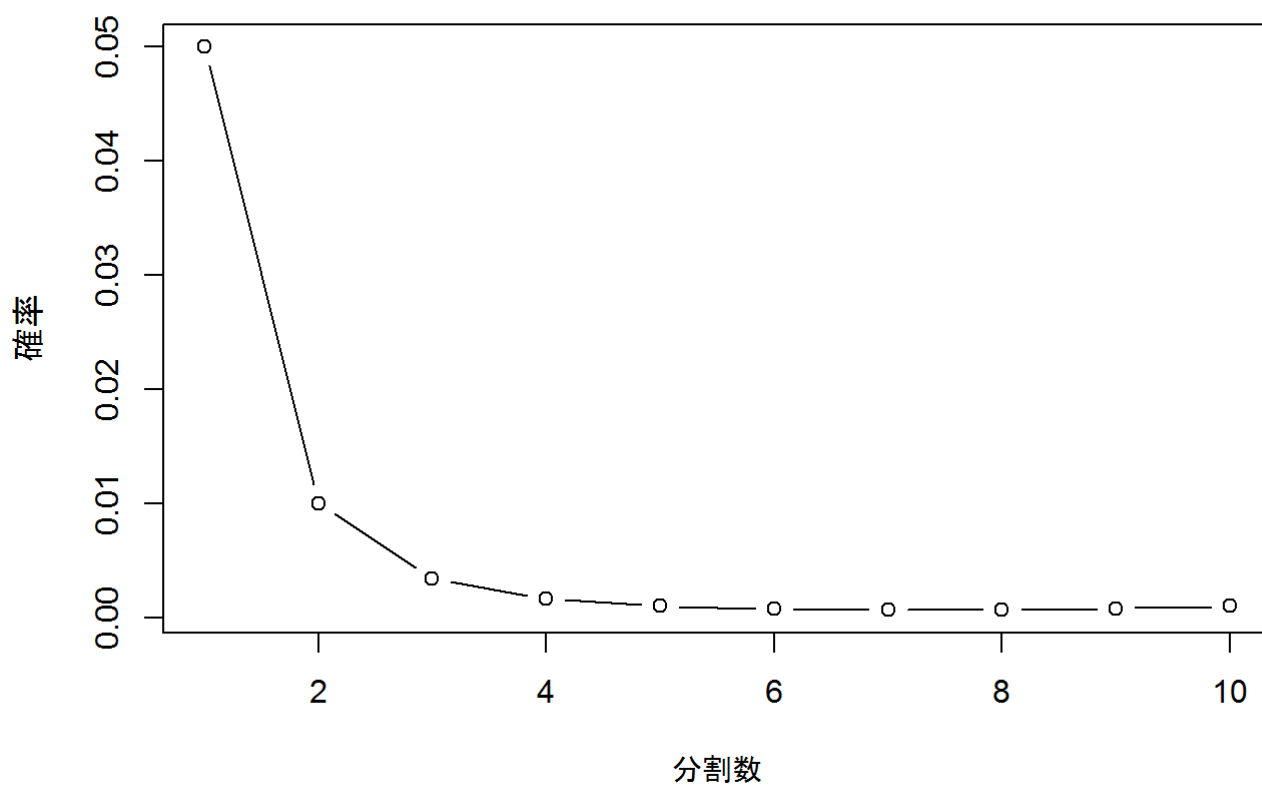
```
n.part <- 1:10
p1 <- 0.05
p.failure <- p1*n.part
plot(n.part, p.failure, xlab="分割数", ylab="確率", main="失敗する確率")
```

失敗する確率



```
plot(n. part, p. failure^n. part, type="b", xlab="分割数", ylab="確率", main="すべて失敗する確率")
```

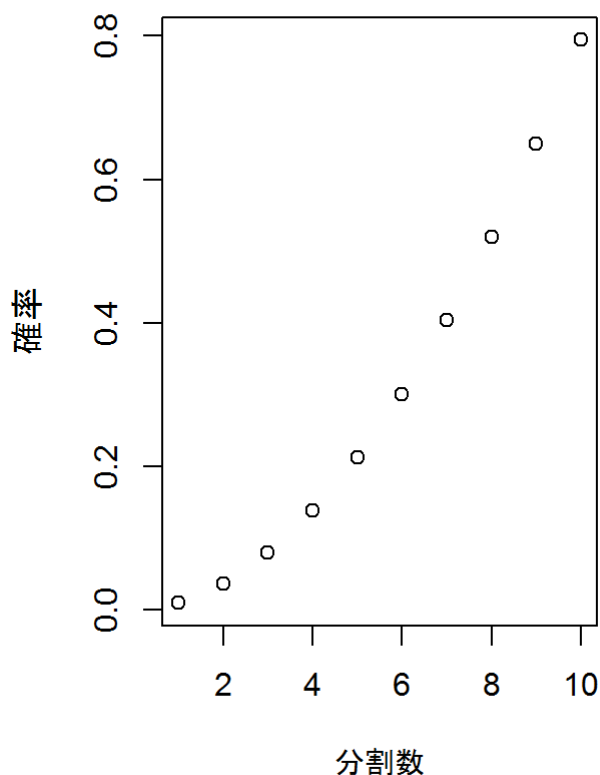
すべて失敗する確率



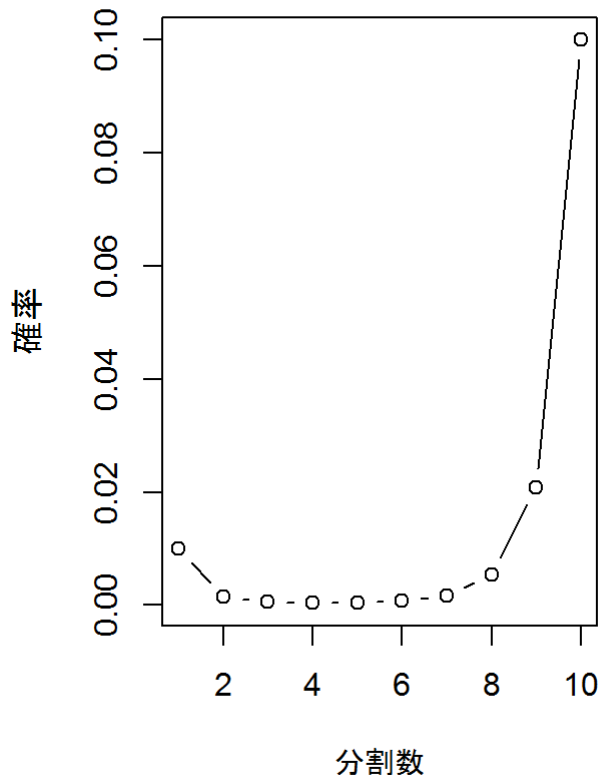
分割に従ってもっとどんどん成功率が下がる(失敗率が上がる)と...

```
n.part <- 1:10
p1 <- 0.01
p.failure <- p1*n.part^1.9
par(mfcol=c(1,2))
plot(n.part, p.failure, xlab="分割数", ylab="確率", main="失敗する確率")
plot(n.part, p.failure^n.part, type="b", xlab="分割数", ylab="確率", main="すべて失敗する確率")
```

失敗する確率



すべて失敗する確率



```
par(mfcol=c(1,1))
```

理論編

分割してもしなくても同じ場合

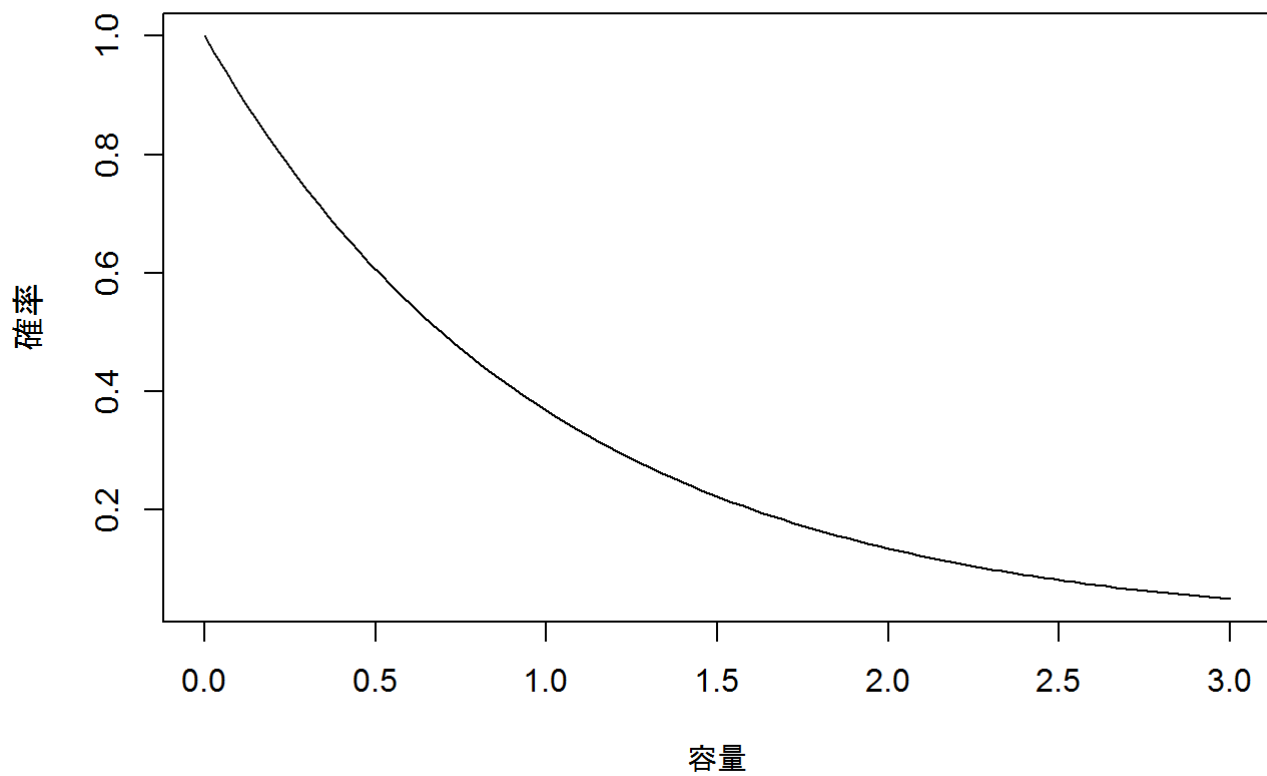
容量 v の減少により、指数関数的に失敗率が下がる

$v=0$ の失敗率が1

$$f(v) = e^{-av}$$

```
v <- seq(from=0, to=3, length=100)
a <- 1
plot(v, exp(-a*v), type="l", xlab="容量", ylab="確率", main="失敗確率")
```

失敗確率



総量 $v=1$

分割してみる

(0.5,0.5) (0.2,0.8) (0.5,0.3,0.2)

```
# utility functions
my.g <- function(x, as, ps=0:(length(as)-1), scale=1) {
  ret <- rep(0, length(x))
  for(i in 1:length(as)) {
    ret <- ret + (as[i]/scale)*x^ps[i]
  }
  ret
}

my.div.g <- function(vs, as, ps=0:(length(as)-1), scale=1, log=FALSE) {
  ret <- 0
  for(i in 1:length(vs)) {
    ret <- ret + my.g(vs[i], as, ps, scale)
  }
  if(!log) {
    ret <- exp(ret)
  }
  return(ret)
}

as <- c(-1)
ps <- c(1)
v <- seq(from=0, to=2, length=101)
scale <- 1
#plot(v, exp(my.g(v, as, ps, scale)), type="l")
```

2分割なら2回とも失敗する確率、 3分割なら3回とも失敗する確率を計算してみる。

```
vs <- c(0.5, 0.5)
my.div.g(vs, as, ps, scale)
```

```
## [1] 0.3678794
```

```
vs <- c(0.8, 0.2)
my.div.g(vs, as, ps, scale)
```

```
## [1] 0.3678794
```

```
vs <- c(0.5, 0.3, 0.2)
my.div.g(vs, as, ps, scale)
```

```
## [1] 0.3678794
```

分割の仕方によらず、すべて失敗する確率は同じ。

なぜなのか？

総量Vの試料をk回に分割する

$$\sum_{i=1}^k v_i = V$$

各回の失敗率

$$f(v) = e^{-av}$$

全回、失敗する確率は、以下の通り、分割の仕方によらずVのみによる

$$\prod_{i=1}^k f(v) = \prod_{i=1}^k e^{-av} = e^{\sum_{i=1}^k -av} = e^{-aV}$$

$$f(v) = e^{-av}$$

という特別な失敗率と容量との関係があるとき、その分割の仕方によらず、全回、失敗する確率は総量によって決まる。

「指数関数」は「掛け算が足し算になる」ので、

「指数関数的に失敗率が決まるとき、分割の仕方によらず、全て失敗する確率は、使用する試料の総量によって決まる」

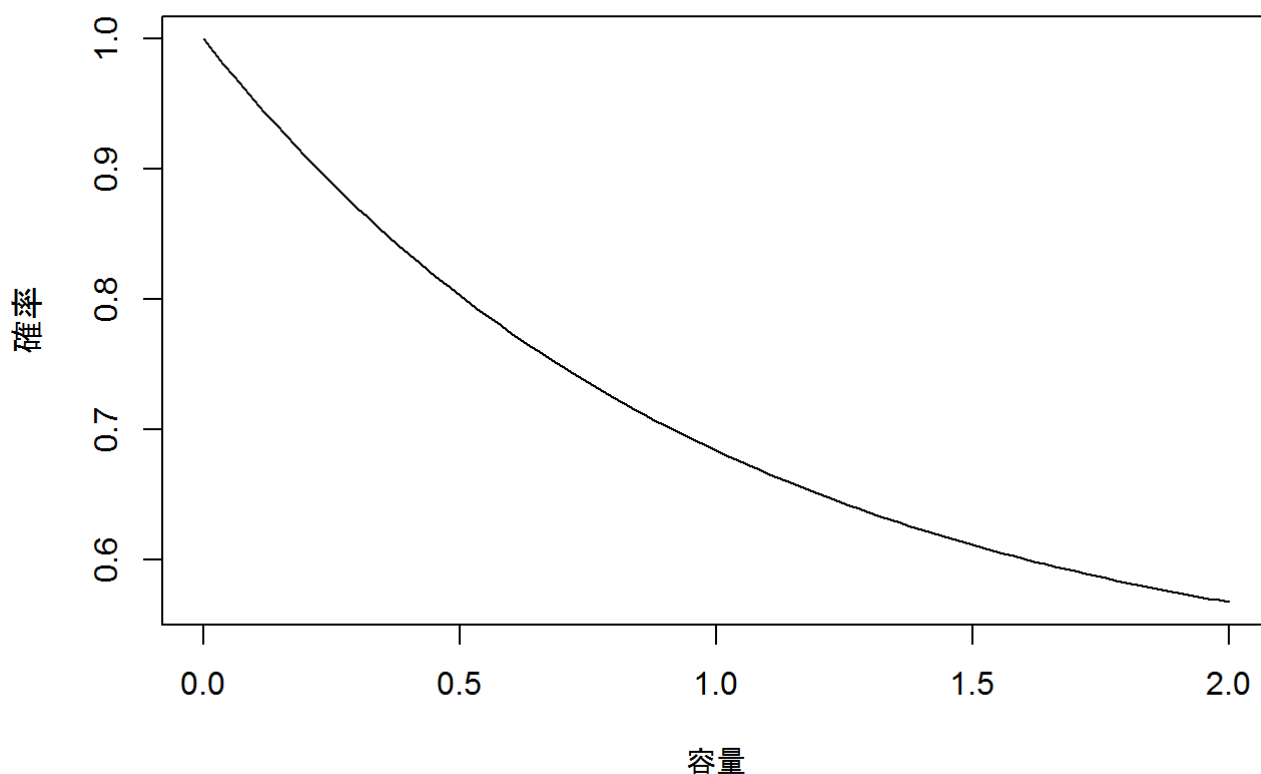
容量非依存性にも失敗するとすると

容量0のときの成功率0、容量が無限大のときの失敗率、 $K/(1+K)$

$$f(v) = \frac{e^{-av} + K}{1 + K}$$

```
K <- 1
plot(v, (exp(-v)+K)/(1+K), type="l", xlab="容量", ylab="確率", main="失敗確率")
```

失敗確率



二等分、三等分、四等分で、「全て失敗する確率」は変わってくる

```
vs <- rep(1/2, 2)
prod((exp(-vs)+K)/(1+K))
```

```
## [1] 0.6452352
```

```
vs <- rep(1/3, 3)
prod((exp(-vs)+K)/(1+K))
```

```
## [1] 0.6322156
```

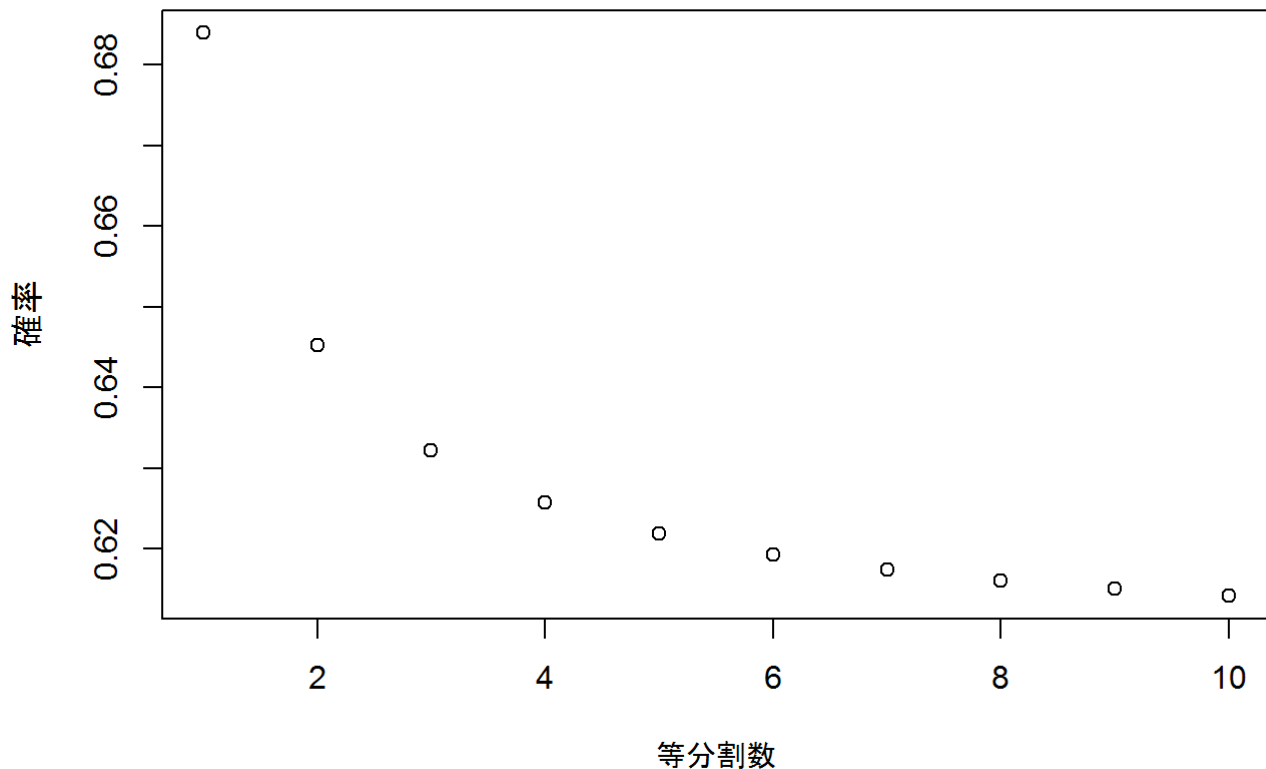
```
vs <- rep(1/4, 4)
prod((exp(-vs)+K)/(1+K))
```

```
## [1] 0.6257333
```

n等分するとき、nと「全て失敗する確率」との関係

```
n.v <- 1:10
out <- rep(0, length(n.v))
for(i in 1:length(n.v)) {
  vs <- rep(1/n.v[i], n.v[i])
  out[i] <- prod((exp(-vs)+K)/(1+K))
}
plot(n.v, out, xlab="等分割数", ylab="確率", main="すべて失敗する確率")
```

すべて失敗する確率



失敗率関数を一般化してみる

指数関数がなかなか良い仕事をするので、指数関数を使う

ただし、失敗率関数を一般化したいので、指数を試料量の関数(特に多項式)で表すことにする

$$f(v) = e^{g(v)}$$

$$g(v) = -av$$

$$g(v) = \sum_{i=1}^k a_i v^{p_i}$$

$g(v)$ が単純な場合

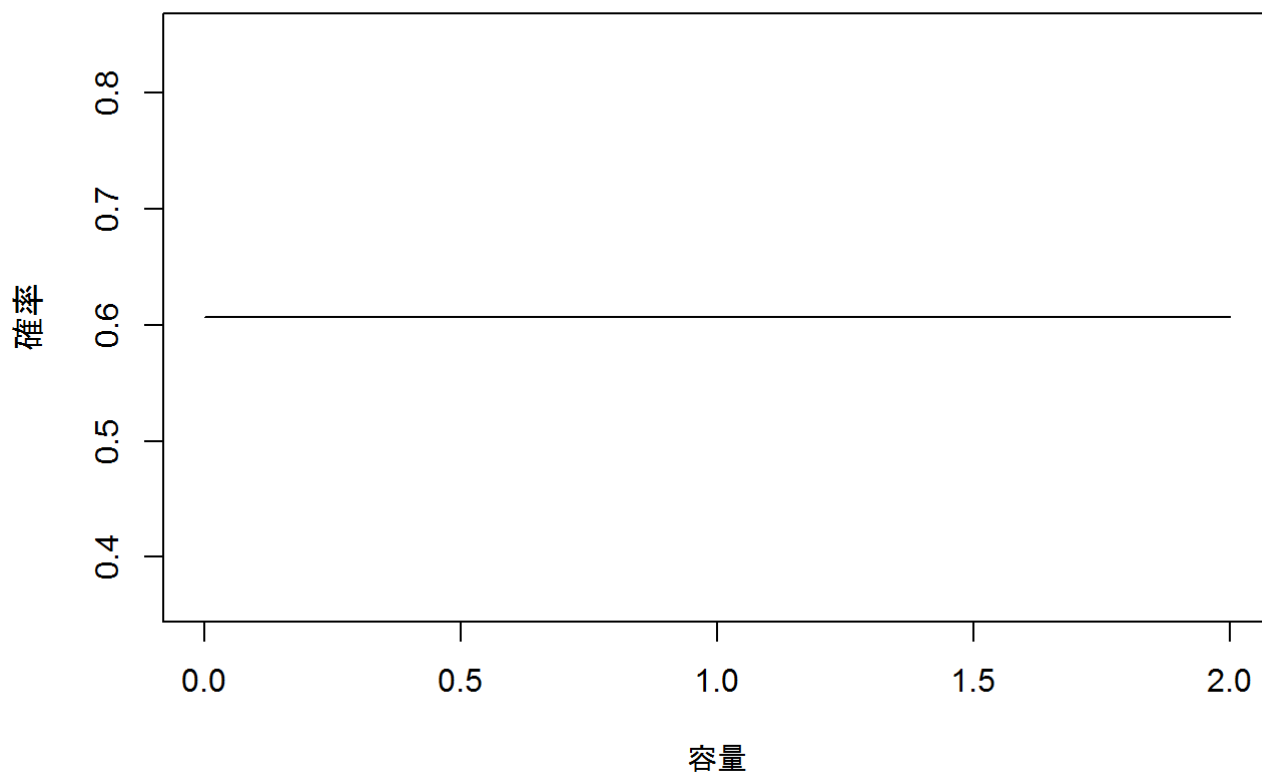
$$g(v) = a \times v^p$$

$p=0$ の場合

$$g(v) = a \times v^0 = a$$

```
as <- c(-0.5)
ps <- c(0)
v <- seq(from=0, to=2, length=101)
scale <- 1
plot(v, exp(my.g(v, as, ps, scale)), type="l", xlab="容量", ylab="確率", main="失敗確率")
```

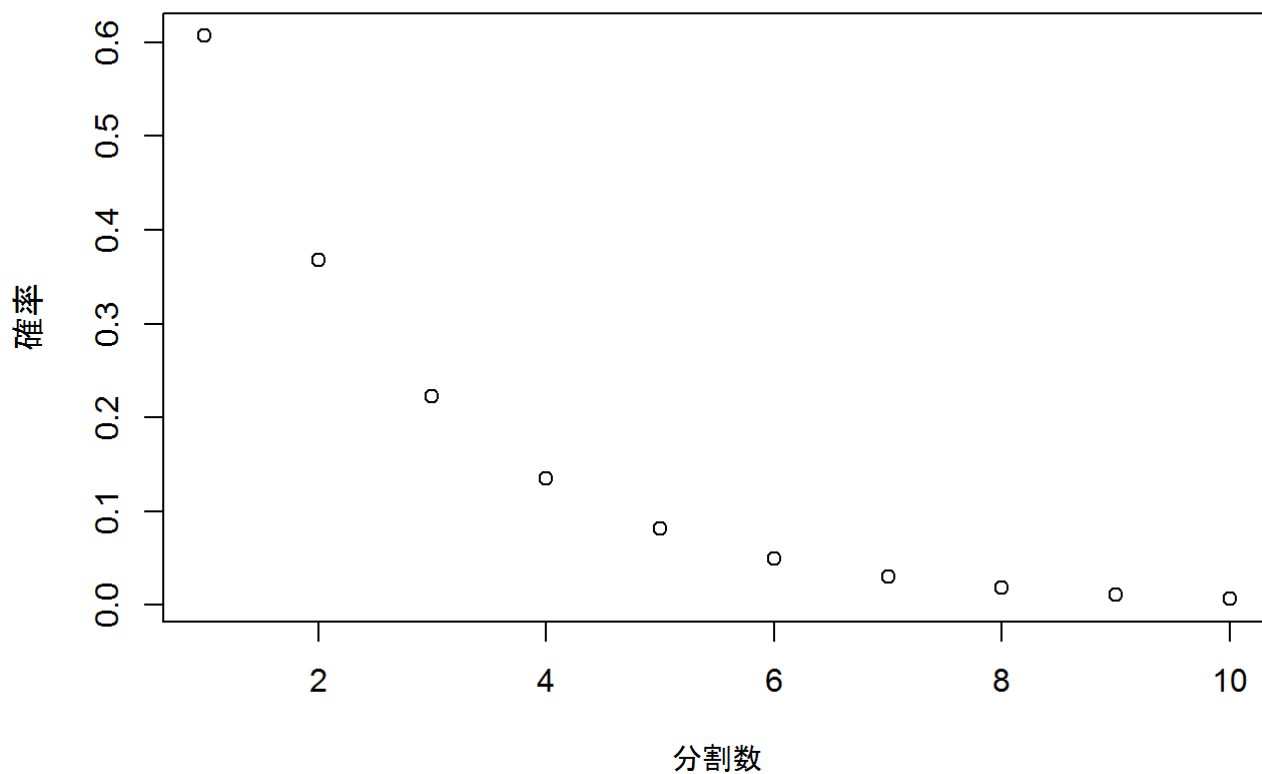
失敗確率



等分割するとしたら

```
n.v <- 1:10
out <- rep(0, length(n.v))
for(i in 1:length(n.v)){
  vs <- rep(1/n.v[i], n.v[i])
  out[i] <- my.div.g(vs, as, ps, scale)
}
plot(n.v, out, xlab="分割数", ylab="確率", main="全て失敗する確率")
```

全て失敗する確率

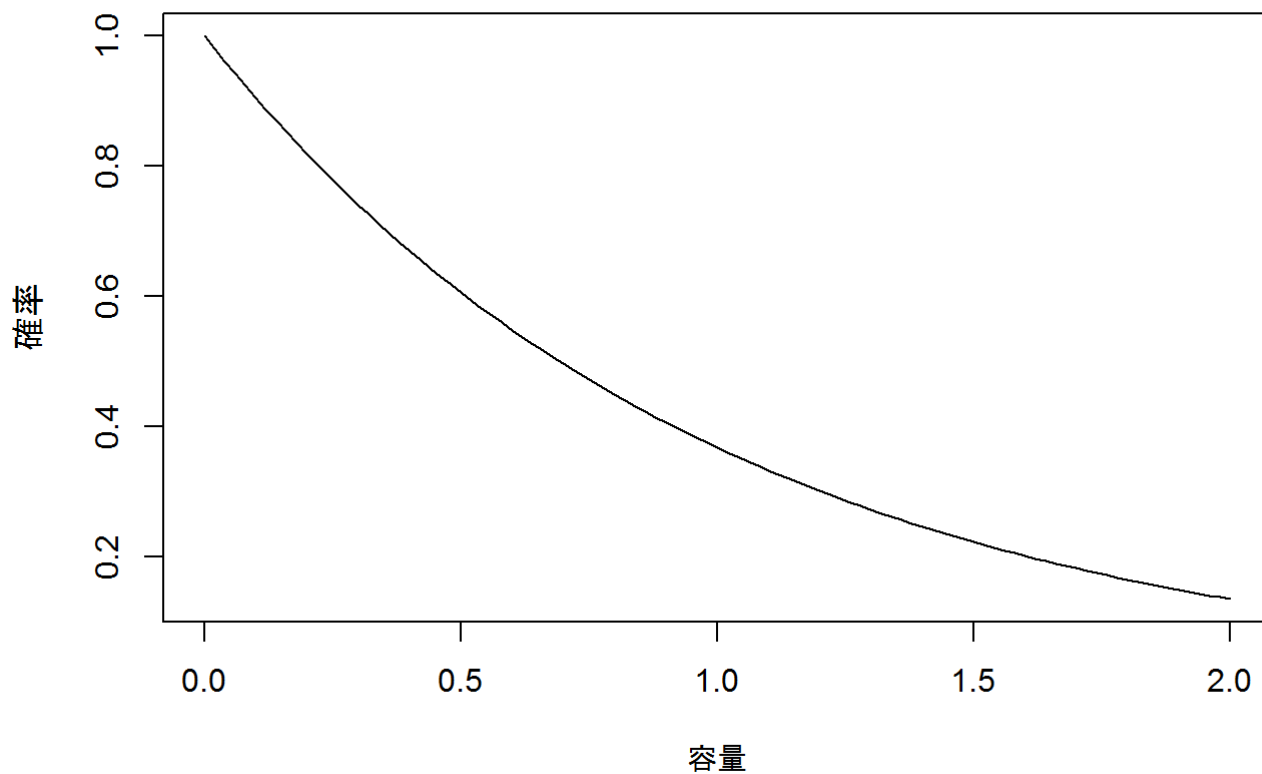


p=1の場合

$$g(v) = a \times v^1 = av$$

```
as <- c(-1)
ps <- c(1)
v <- seq(from=0, to=2, length=101)
scale <- 1
plot(v, exp(my.g(v, as, ps, scale)), type="l", xlab="容量", ylab="確率", main="失敗確率")
```

失敗確率

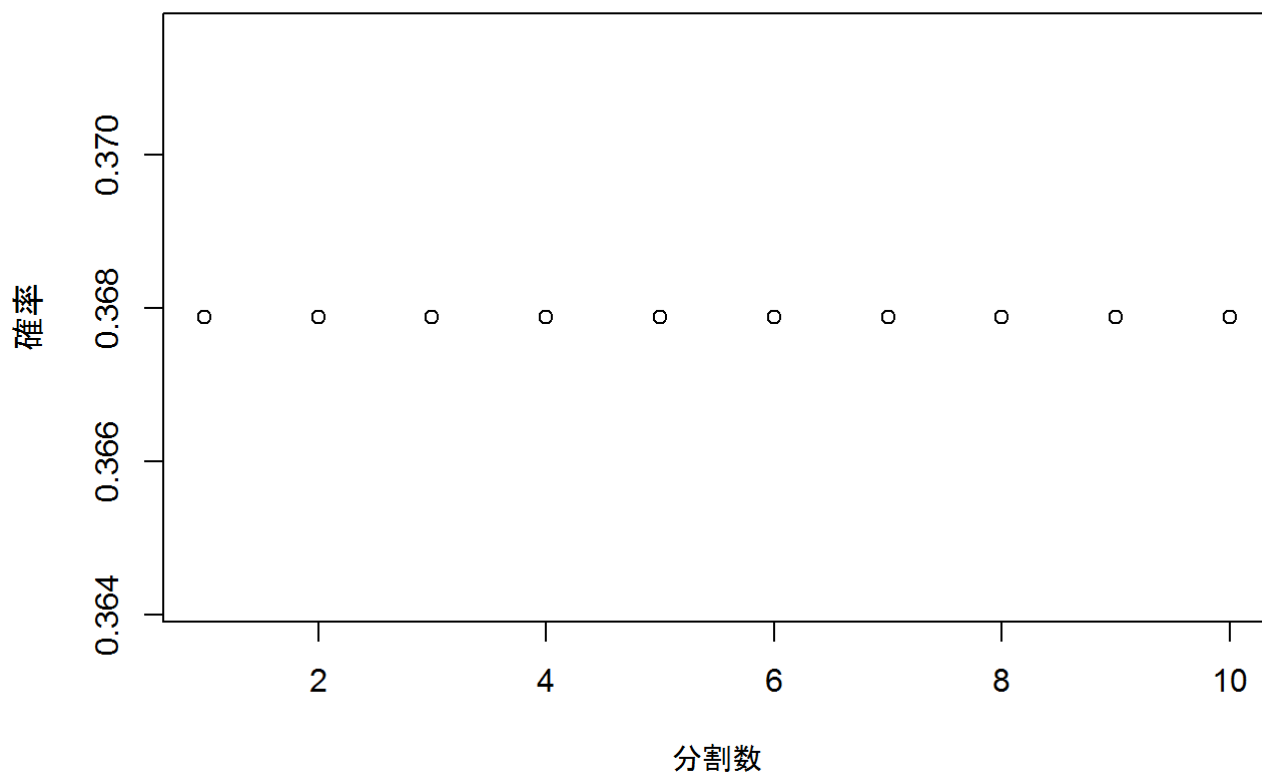


等分割する

これは、「失敗率関数が単純な指数関数の場合」 = 「全部失敗する確率は分割の仕方によらない」

```
n.v <- 1:10
out <- rep(0, length(n.v))
for (i in 1:length(n.v)) {
  vs <- rep(1/n.v[i], n.v[i])
  out[i] <- my.div.g(vs, as, ps, scale)
}
plot(n.v, out, xlab="分割数", ylab="確率", main="少なくとも1回は成功する確率")
```

少なくとも1回は成功する確率

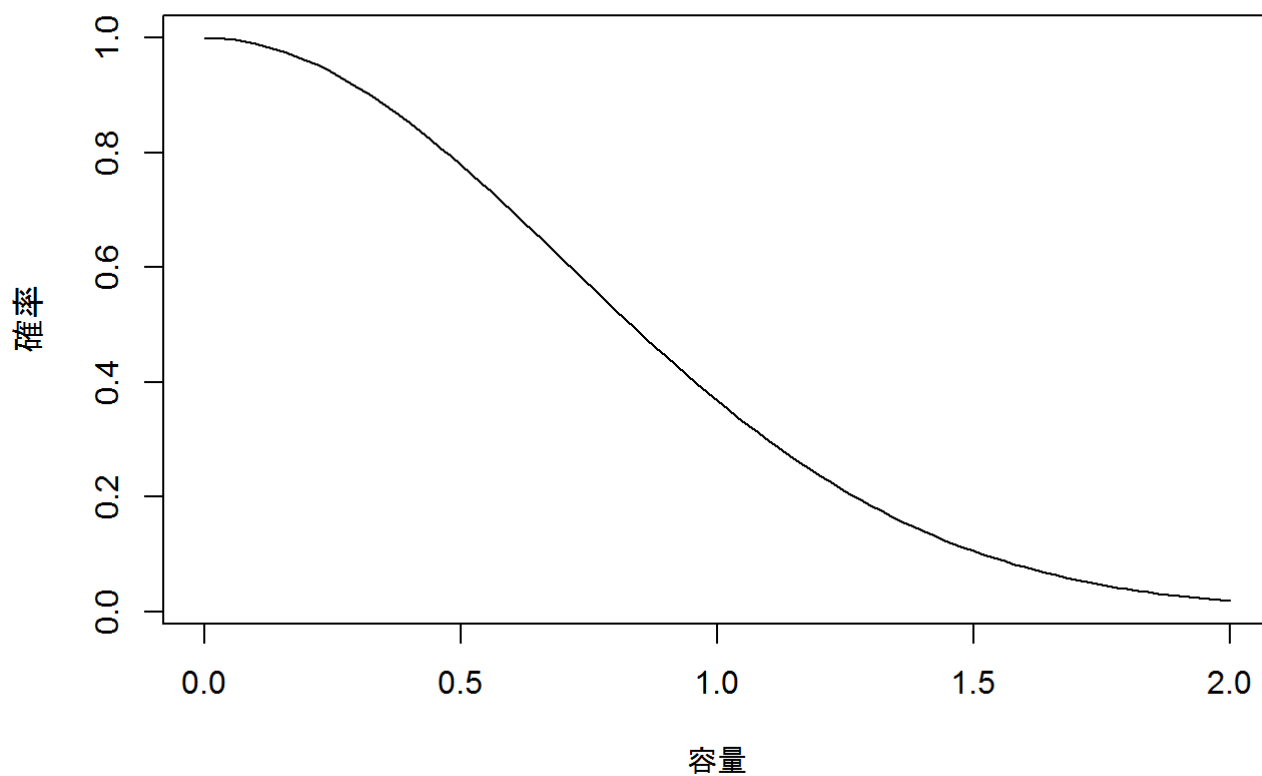


p=2の場合

$$g(v) = a \times v^2 = av^2$$

```
as <- c(-1)
ps <- c(2)
v <- seq(from=0, to=2, length=101)
scale <- 1
plot(v, exp(my.g(v, as, ps, scale)), type="l", xlab="容量", ylab="確率", main="失敗確率")
```

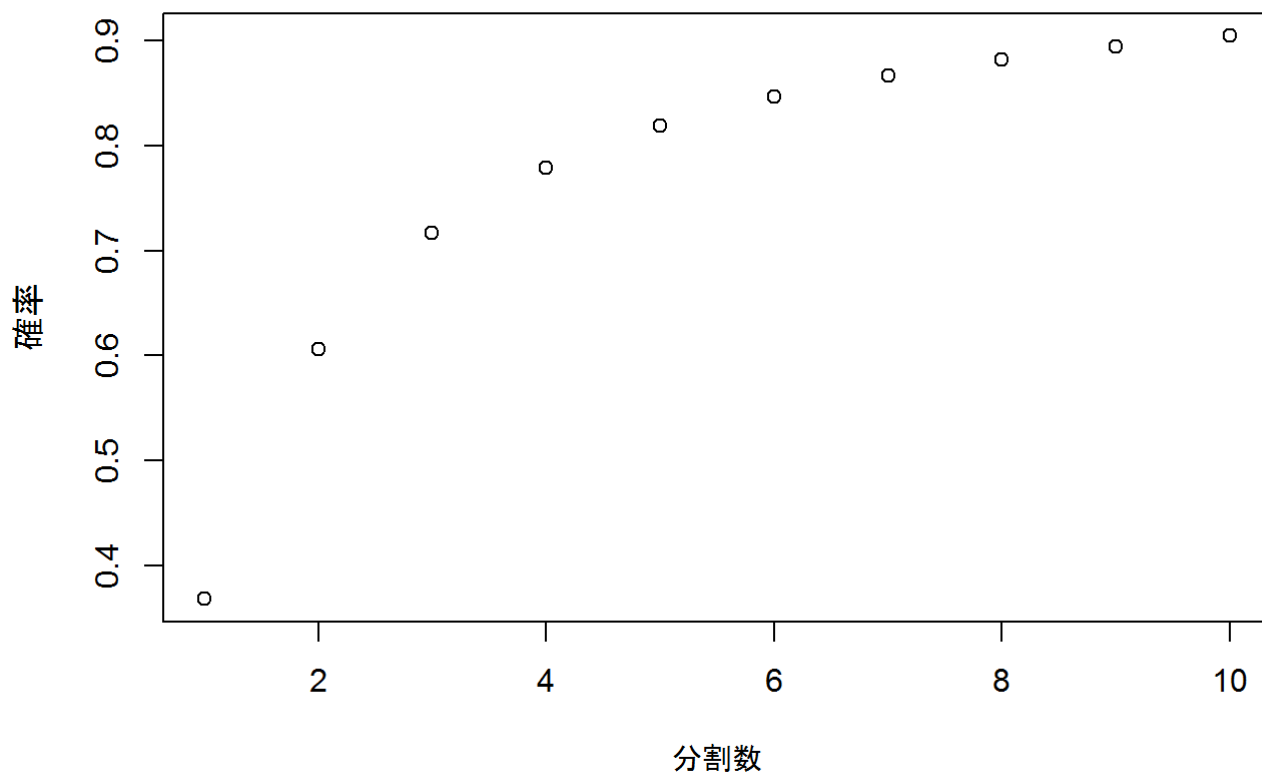
失敗確率



等分割する

```
n.v <- 1:10
out <- rep(0, length(n.v))
for(i in 1:length(n.v)) {
  vs <- rep(1/n.v[i], n.v[i])
  out[i] <- my.div.g(vs, as, ps, scale)
}
plot(n.v, out, xlab="分割数", ylab="確率", main="すべて失敗する確率")
```

すべて失敗する確率

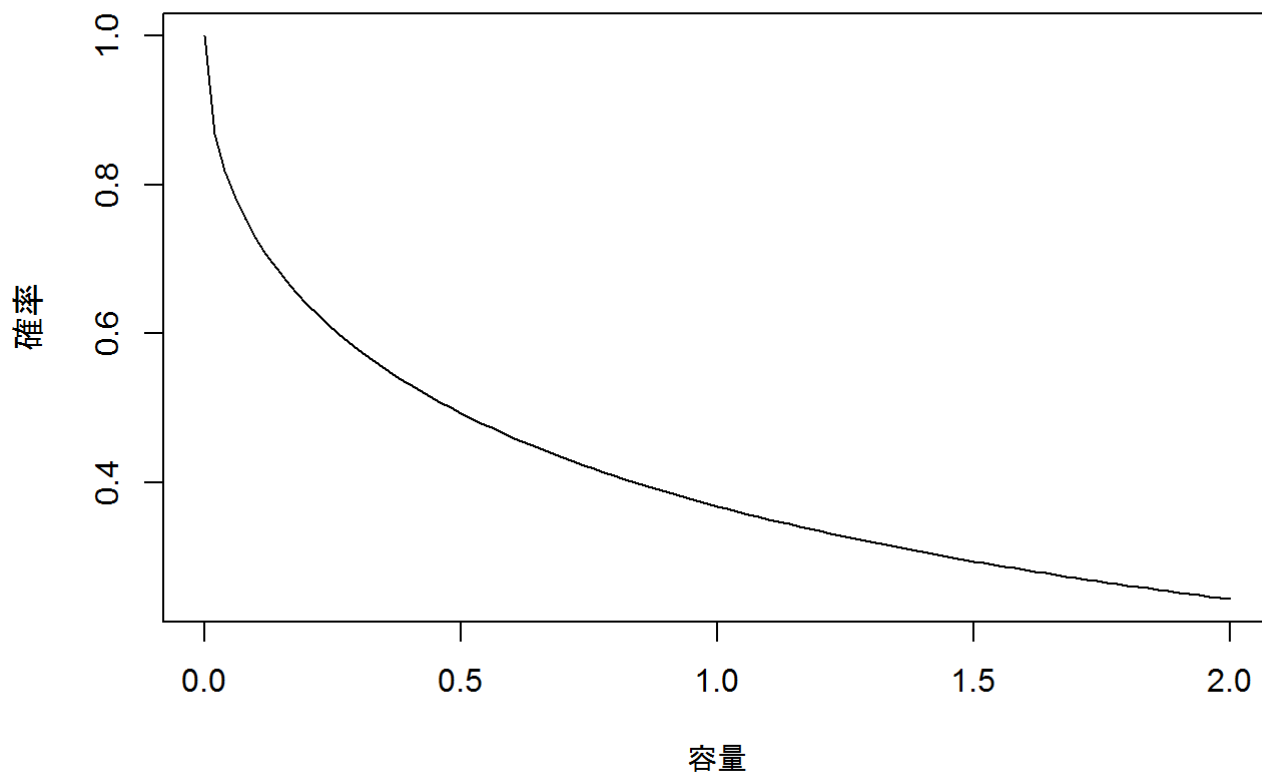


p=0.5の場合

$$g(v) = a \times v^{0.5}$$

```
as <- c(-1)
ps <- c(0.5)
v <- seq(from=0, to=2, length=101)
scale <- 1
plot(v, exp(my.g(v, as, ps, scale)), type="l", xlab="容量", ylab="確率", main="失敗確率")
```

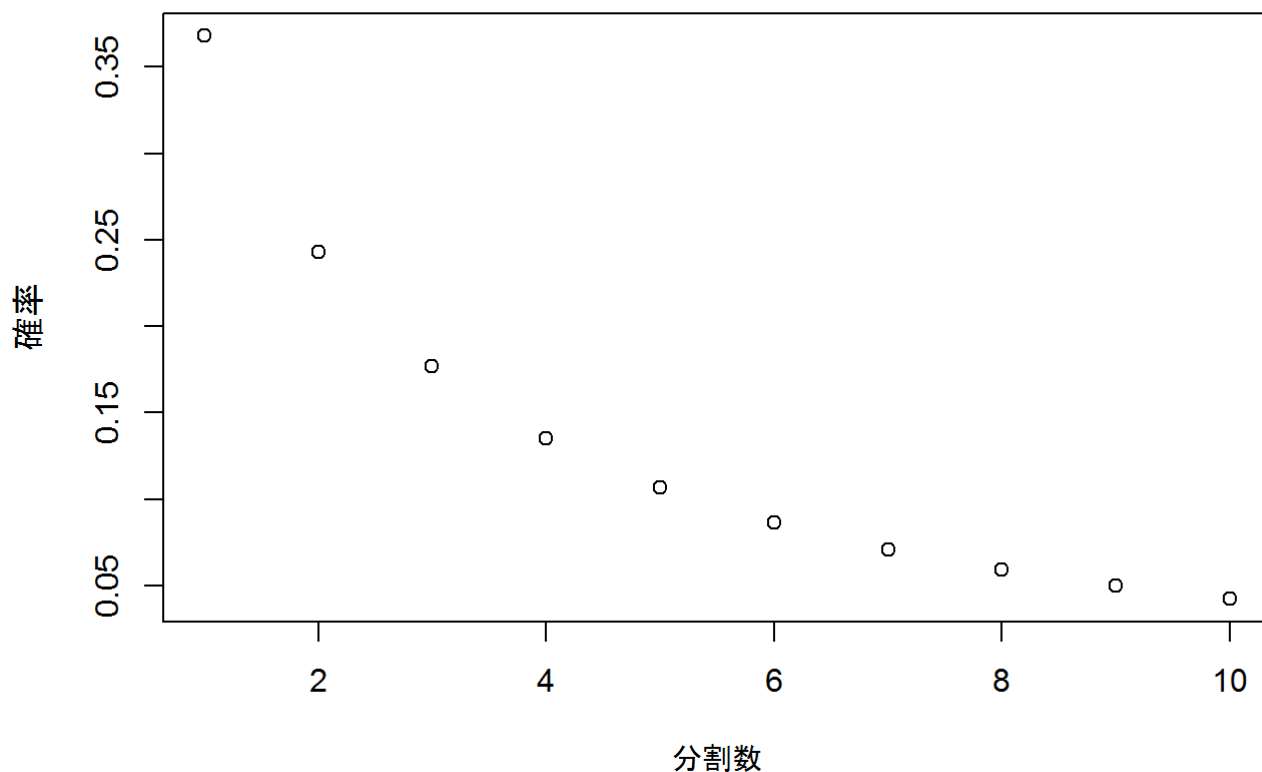
失敗確率



等分割する

```
n.v <- 1:10
out <- rep(0, length(n.v))
for(i in 1:length(n.v)) {
  vs <- rep(1/n.v[i], n.v[i])
  out[i] <- my.div.g(vs, as, ps, scale)
}
plot(n.v, out, xlab="分割数", ylab="確率", main="すべて失敗する確率")
```

すべて失敗する確率



1,2,3分割

分割数を 1, 2, 3 のいずれかにする。

分割はいろいろな割合にする。

$g(v) = a \times v^2$ の場合

$$g(v) = a \times v^2 = av$$

分割する方がよく、分割数が多い方がよく、均等分割が一番よい。

```
library(MCMCpack)
```

```
## Loading required package: coda
```

```
## Loading required package: MASS
```

```
## ##
## ## Markov Chain Monte Carlo Package (MCMCpack)
```

```
## ## Copyright (C) 2003–2017 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park
```

```
## ##
## ## Support provided by the U.S. National Science Foundation
```

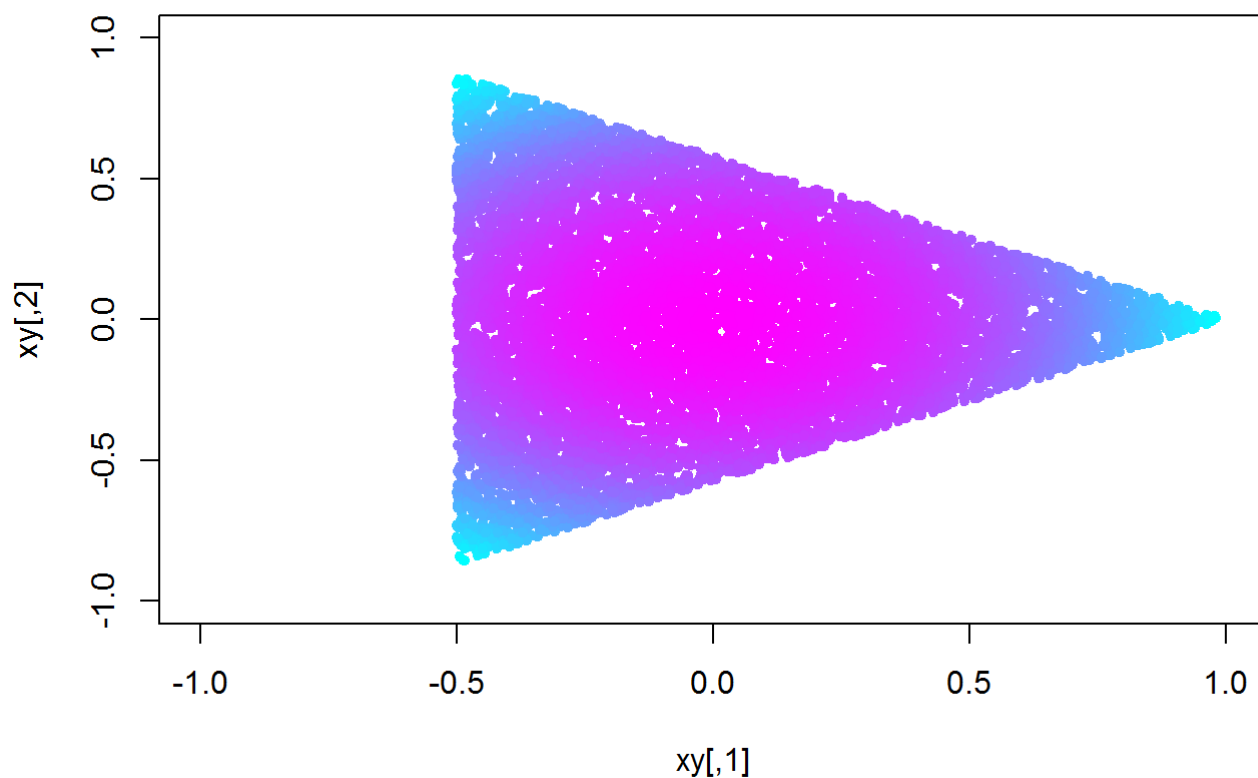
```
## ## (Grants SES-0350646 and SES-0350613)
## ##
```

```
library(mwaytable)
n.pt <- 10^4
Vs <- rdirichlet(n.pt, rep(1, 3))
cv <- CategoryVector(3)
xy <- Vs %*% cv
```

```
as <- c(-1)
ps <- c(2)
v <- seq(from=0, to=2, length=101)
scale <- 1

out <- rep(0, n.pt)
for(i in 1:n.pt){
  vs <- Vs[i,]
  out[i] <- my.div.g(vs, as, ps, scale, log=TRUE)
}

out.st <- (out-min(out))/(max(out)-min(out))
plot(xy, pch=20, col=rgb(out.st, 1-out.st, 1), xlim = c(-1, 1), ylim=c(-1, 1))
```



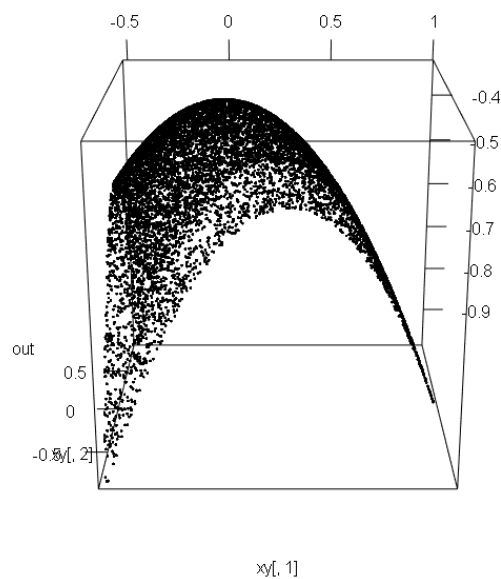
```
library(rgl)
```

```
## Warning: package 'rgl' was built under R version 3.2.5
```

```
library(knitr)
```

```
knit_hooks$set(rgl = hook_rgl)
```

```
plot3d(xy[, 1], xy[, 2], out)
```



$g(v) = a \times v^{0.5}$ の場合

$$g(v) = a \times v^{0.5}$$

の場合は

分割すればするほど悪くなる。同じ分割数なら、均等分割が最悪。

```
library(MCMCpack)
library(mwaytable)
n.pt <- 10^4
Vs <- rdirichlet(n.pt, rep(1, 3))
cv <- CategoryVector(3)
xy <- Vs %*% cv
```

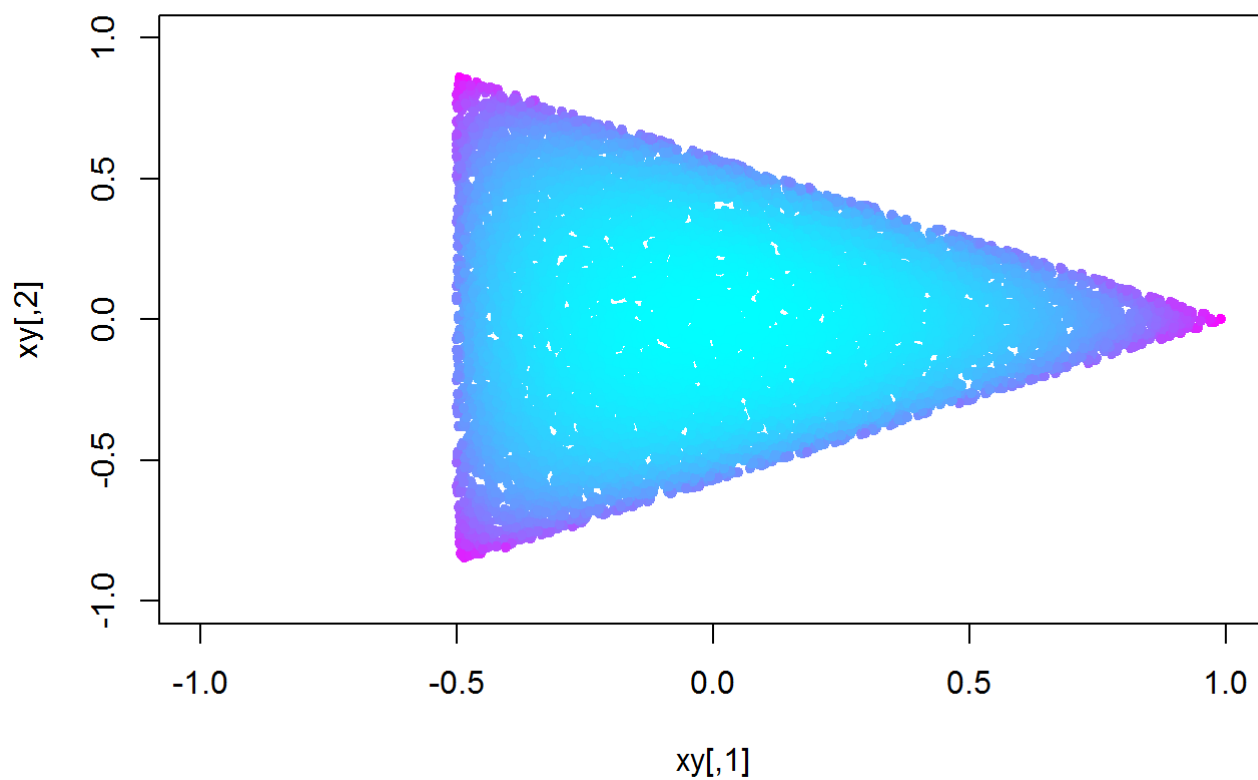
```

as <- c(-1)
ps <- c(0.5)
v <- seq(from=0, to=2, length=101)
scale <- 1

out <- rep(0, n.pt)
for(i in 1:n.pt){
  vs <- Vs[i,]
  out[i] <- my.div.g(vs, as, ps, scale, log=TRUE)
}

out.st <- (out-min(out))/(max(out)-min(out))
plot(xy, pch=20, col=rgb(out.st, 1-out.st, 1), xlim = c(-1, 1), ylim=c(-1, 1))

```



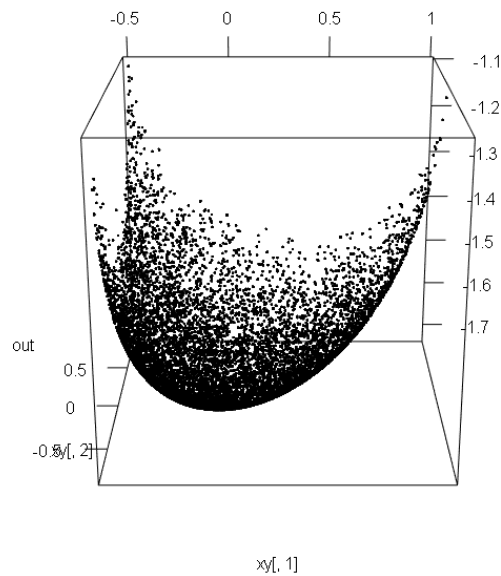
```

library(rgl)
library(knitr)

```

```
knit_hooks$set(rgl = hook_rgl)
```

```
plot3d(xy[, 1], xy[, 2], out)
```



$g(v)$ が複雑になると分割する方がよいか、しない方がよいかは単純ではない

$$g(v) = -v^{0.5} - v^2$$

```
Vs <- rdirichlet(n.pt, rep(1, 3))
cv <- CategoryVector(3)
xy <- Vs %*% cv
```

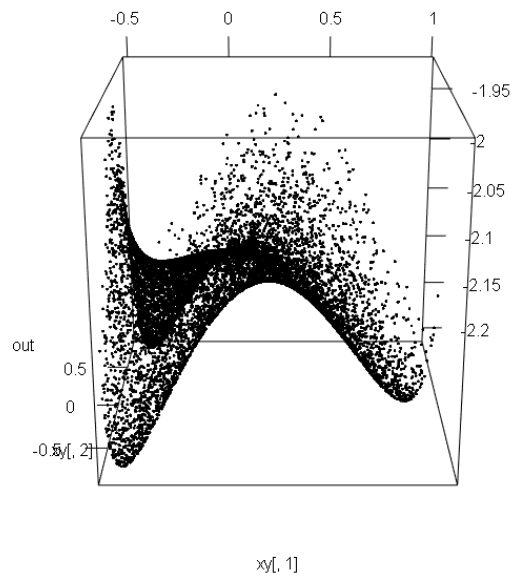
```
as <- c(-1, -1)
ps <- c(0.5, 2)
v <- seq(from=0, to=2, length=101)
scale <- 1

out <- rep(0, n.pt)
for(i in 1:n.pt){
  vs <- Vs[i,]
  out[i] <- my.div.g(vs, as, ps, scale, log=TRUE)
}
```

```
library(rgl)
library(knitr)
```

```
knit_hooks$set(rgl = hook_rgl)
```

```
plot3d(xy[, 1], xy[, 2], out)
```



総量を変えても、 $g(v)$ が単純な式なら、分割の良し悪しは変わらない。

$$\prod_{i=1}^k e^{g(v_i)} = e^{\sum_{i=1}^k g(v_i)} = e^{\sum_{i=1}^k a v_i^p}$$

単純な場合

$$\sum_{i=1}^k a v_i^p$$

総量をw倍する

$$\sum_{i=1}^k a (w v_i)^p = \sum_{i=1}^k a w^p v_i^p = w^p \sum_{i=1}^k a v_i^p$$

```
Vs <- rdirichlet(n.pt, rep(1, 3)) * 5
cv <- CategoryVector(3)
xy <- Vs %*% cv
```

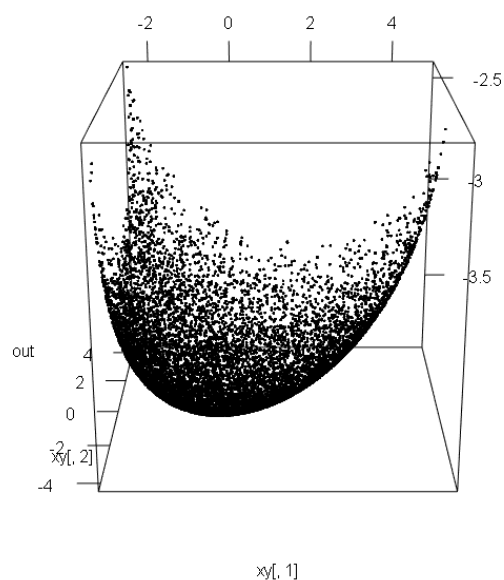
```
as <- c(-1)
ps <- c(0.5)
v <- seq(from=0, to=2, length=101)
scale <- 1

out <- rep(0, n.pt)
for(i in 1:n.pt){
  vs <- Vs[i,]
  out[i] <- my.div.g(vs, as, ps, scale, log=TRUE)
}
```

```
library(rgl)
library(knitr)
```

```
knit_hooks$set(rgl = hook_rgl)
```

```
plot3d(xy[, 1], xy[, 2], out)
```



複雑な場合

$$g(v) = -v^{0.5} - v^2$$

$g(v)$ が複雑になると総量の影響を受ける

```
Vs <- rdirichlet(n.pt, rep(1, 3))
Vs2 <- Vs * 5
cv <- CategoryVector(3)
xy <- Vs %*% cv
xy2 <- Vs2 %*% cv
```

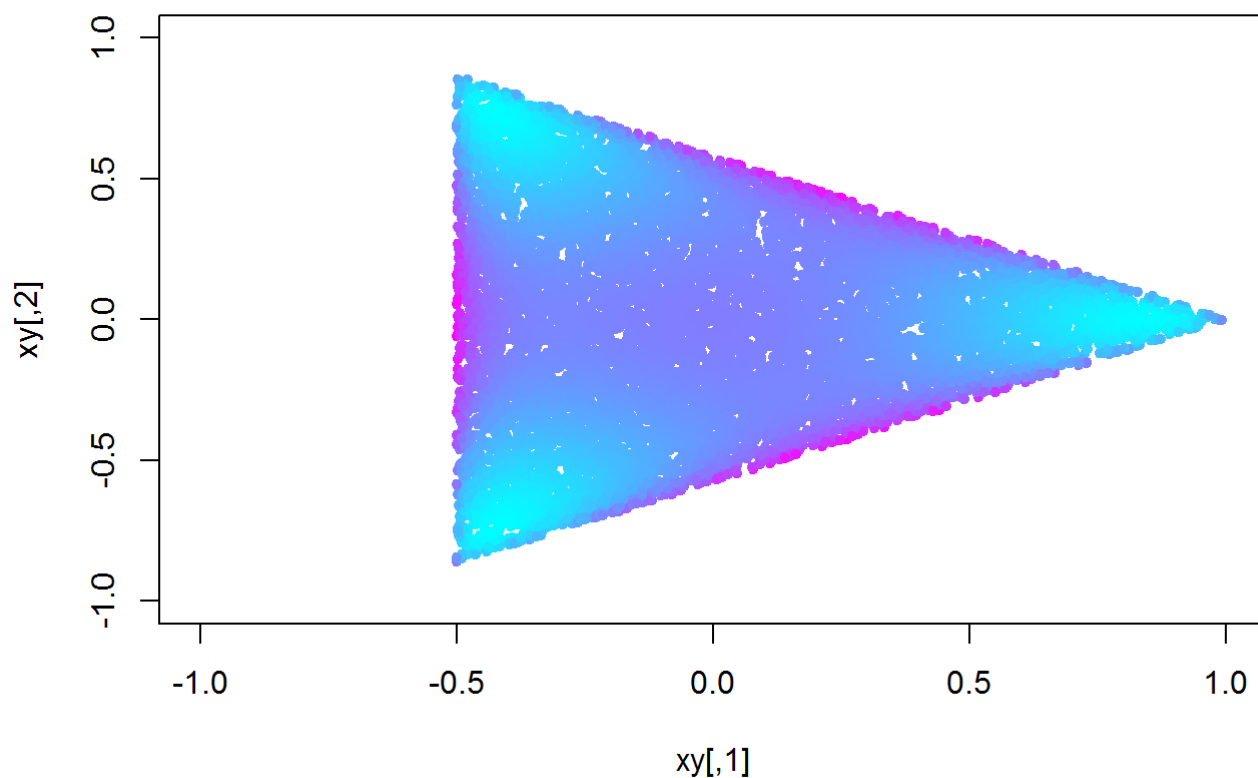
```

as <- c(-1, -1)
ps <- c(0.5, 2)
v <- seq(from=0, to=2, length=101)
scale <- 1

out <- rep(0, n.pt)
out2 <- out
for(i in 1:n.pt){
  vs <- Vs[i,]
  out[i] <- my.div.g(vs, as, ps, scale, log=TRUE)
  vs <- Vs2[i,]
  out2[i] <- my.div.g(vs, as, ps, scale, log=TRUE)
}

out.st <- (out-min(out))/(max(out)-min(out))
plot(xy, pch=20, col=rgb(out.st, 1-out.st, 1), xlim = c(-1, 1), ylim=c(-1, 1))

```



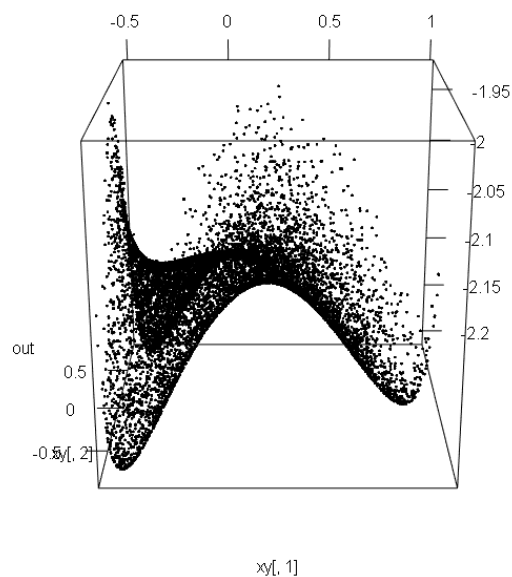
```

library(rgl)
library(knitr)

```

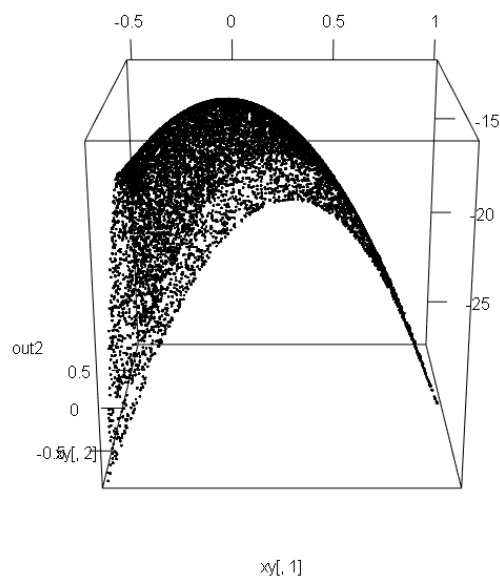
```
knit_hooks$set(rgl = hook_rgl)
```

```
plot3d(xy[, 1], xy[, 2], out)
```



```
knit_hooks$set(rgl = hook_rgl)
```

```
plot3d(xy[, 1], xy[, 2], out2)
```



理論編 まとめ

成功率が試料の量の関数であるとする。

成功率が試料の量に関して指数関数で決まるとき、分割試料のうち少なくとも1つで成功する確率は、分割の仕方によらず一定である。

成功率が試料の量の p 乗に関して指数関数で決まるとき、 p が1より大のときは、分割すればするほどよく、分割は均等分割がよい

特に $p=0$ のとき(試料の量によらずに成功率が決まるとき)は、 $p<1$ の特殊な場合であり、分割すればするほどよい

逆に p が1より小のときは、分割しない方がよく、同じ分割数ならば、分割を均等から遠ざけるのがよい

以上は、成功率が $e^{g(v)}$ と表すとしたときに $g(v)$ が単項式になる場合である。この場合は、どのような分割がよいかは、 $g(v)$ によって決まり、試料の総量が分割の良し悪しに影響しない

成功率が上記のパターンで決まらないときというのは、成功率が試料量の関数で決まるが、それを $e^{g(v)}$ と表したときに、 $g(V)$ が試料量の単項式ではない場合である。そのような場合は、最善分割は単純には決まらない。

また、 $g(v)$ が単項式でない場合は、最善分割は総量によっても変化する

実践編

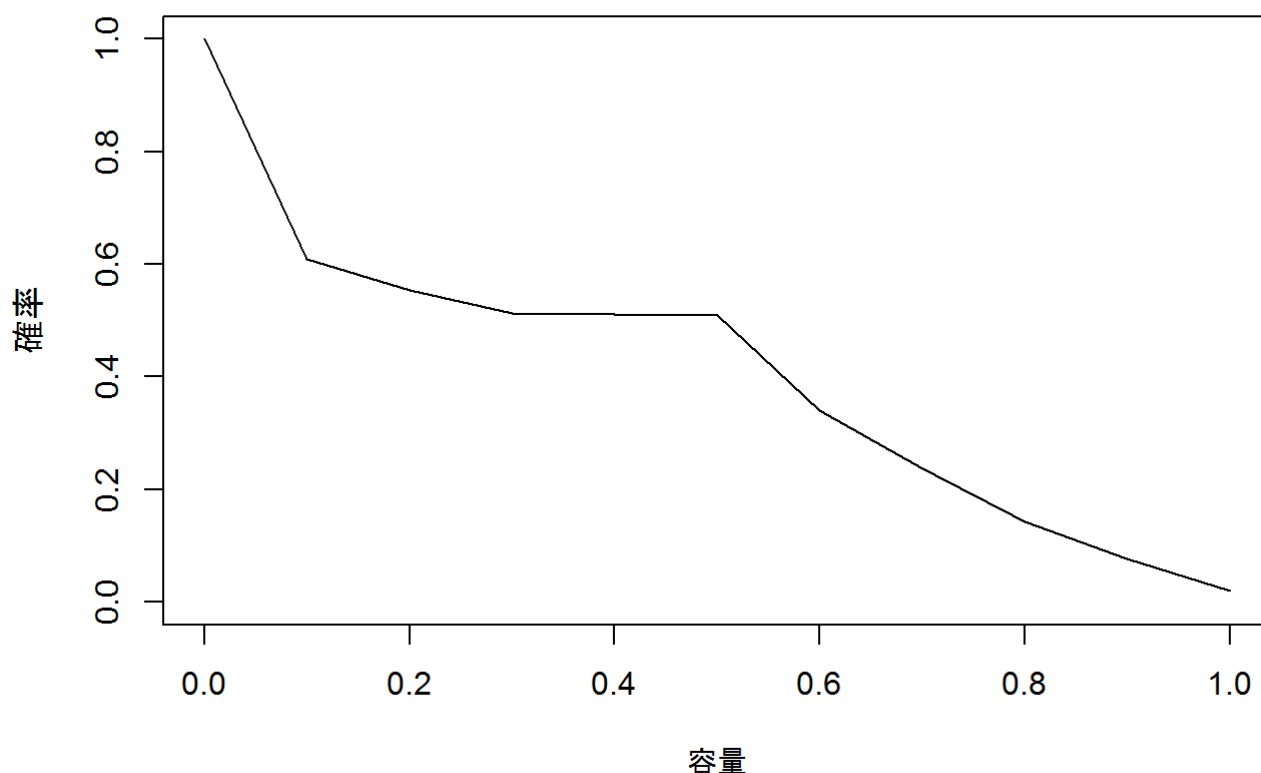
試料総量がわかっている場合

当該実験の成功率特性を実験的に推定することで、最適な分割を決めることができる

こんな計量線実験

```
n.obs <- 10
v <- seq(from=0, to=1, length=n.obs+1)
#tmp <- sort(rdirichlet(1, rep(0.2, n.obs))*0.98, decreasing=TRUE)
tmp <- c(sort(rdirichlet(1, rep(0.5, n.obs/2))/2, decreasing=TRUE), sort(rdirichlet(1, rep(4, n.obs/2))/2, decreasing=TRUE))*0.98
p <- 1-c(0, cumsum(tmp))
plot(v, p, type="l", ylim=c(0, 1), xlab="容量", ylab="確率", main="失敗確率")
```

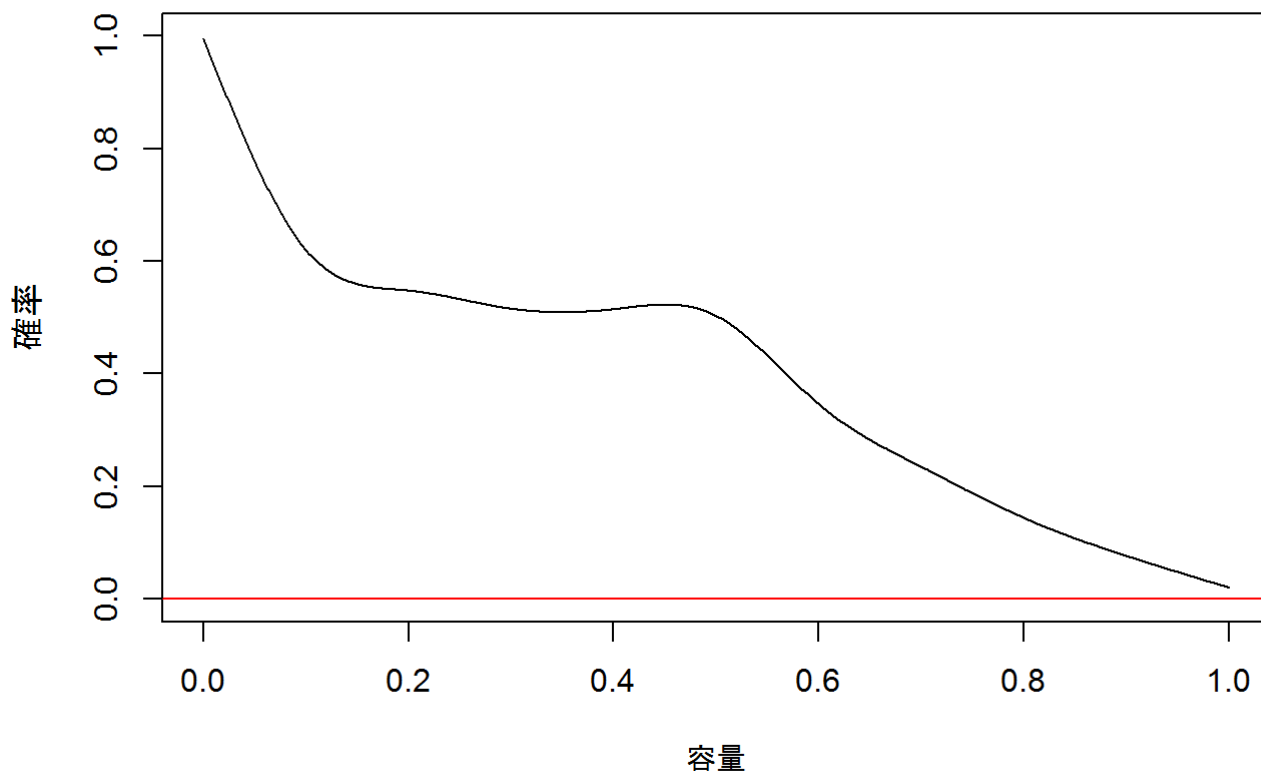
失敗確率



単純に線形補間することにする。

```
sp <- smooth.spline(x=v, y=p, df=length(v)-1)
vv <- seq(from=0, to=1, length=1000)
plot(predict(sp, vv), type="l", ylim=c(0, 1), xlab="容量", ylab="確率", main="失敗確率(補間)")
abline(h=0, col=2)
```

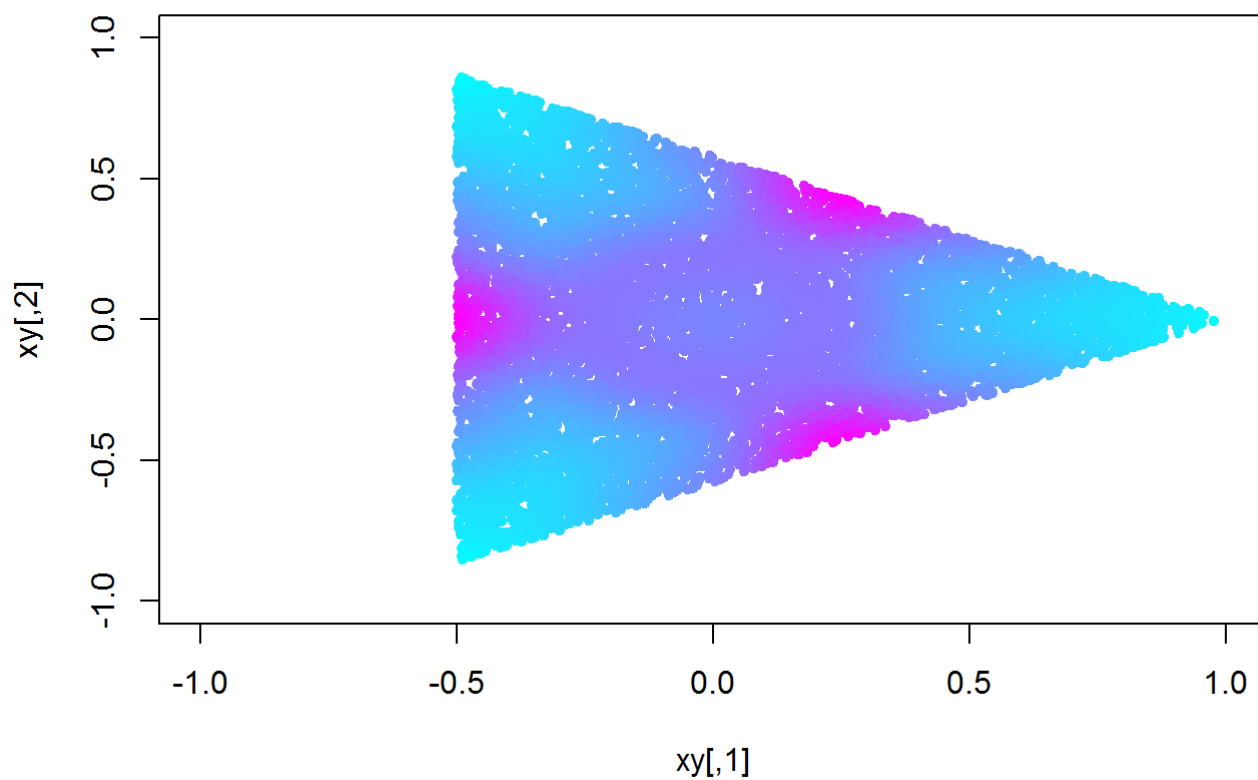
失敗確率(補間)



```
n.pt <- 10^4
Vs <- rdirichlet(n.pt, rep(1, 3))
cv <- CategoryVector(3)
xy <- Vs %*% cv

out <- rep(0, n.pt)
for(i in 1:n.pt){
  vs <- Vs[i,]
  tmp <- predict(sp, vs)
  #print(tmp)
  out[i] <- prod(tmp[[2]])
}
```

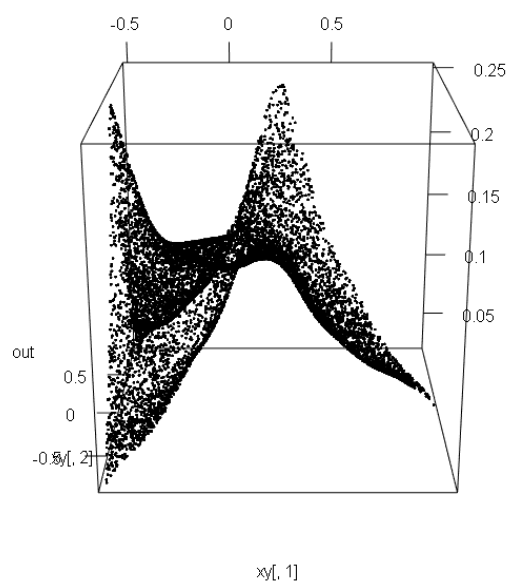
```
out.st <- (out-min(out))/(max(out)-min(out))
plot(xy, pch=20, col=rgb(out.st, 1-out.st, 1), xlim = c(-1, 1), ylim=c(-1, 1))
```



```
library(rgl)
library(knitr)
```

```
knit_hooks$set(rgl = hook_rgl)
```

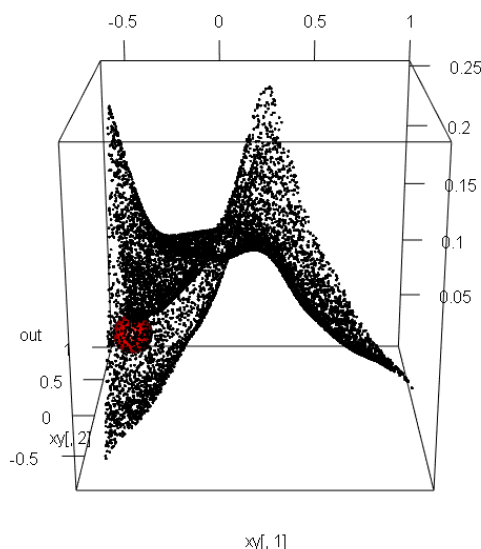
```
plot3d(xy[, 1], xy[, 2], out)
```



```
plot3d(xy[, 1], xy[, 2], out)
min.pt <- which(out==min(out))
print(Vs[min.pt, ])
```

```
## [1] 0.0060782138 0.9933327391 0.0005890472
```

```
spheres3d(c(xy[min.pt, ], out[min.pt]), col=2, radius=0.1)
```



試料総量が不明な場合

試料総量を推定するために、試料の一部を使ってもよい場合

試料総量を推定するために、一部の試料を使用することが可能であれば、総量を推定し、その下で、「試料総量がわかっている場合」に照らして、最適な分割を決めることができる

※ 割愛

試料総量を推定するために、試料の一部を使えないかもしれない場合

これが希少試料問題である。

試料総量が不明であるということを 試料総量に事前分布を入れる、ということにする。

たとえば、0-1の一様分布としてみれば、その事前分布のもとで、分割がもたらす、「成功」分布が得られる。

```
n.r <- 100
prior <- runif(n.r)
n.pt <- 10^4
Vs <- rdirichlet(n.pt, rep(1, 3))
cv <- CategoryVector(3)
xy <- Vs %*% cv

out <- matrix(0, n.pt, n.r)
for(i in 1:n.pt){
  vs <- Vs[i,]
  for(j in 1:n.r){
    tmp <- predict(sp, vs*prior[j])
    #print(tmp)
    out[i, j] <- prod(tmp[[2]])
  }
}
```

すべて失敗する確率の分布をいくつかの分割の場合で表示する

```
par(mfcol=c(3, 3))
s <- sample(1:n.pt, 9)
for(i in s){
  hist(out[i,])
  print(Vs[i,])
}
```

```
## [1] 0.1089105 0.3413615 0.5497280
```

```
## [1] 0.02590158 0.12889944 0.84519898
```

```
## [1] 0.4122318 0.2424043 0.3453638
```

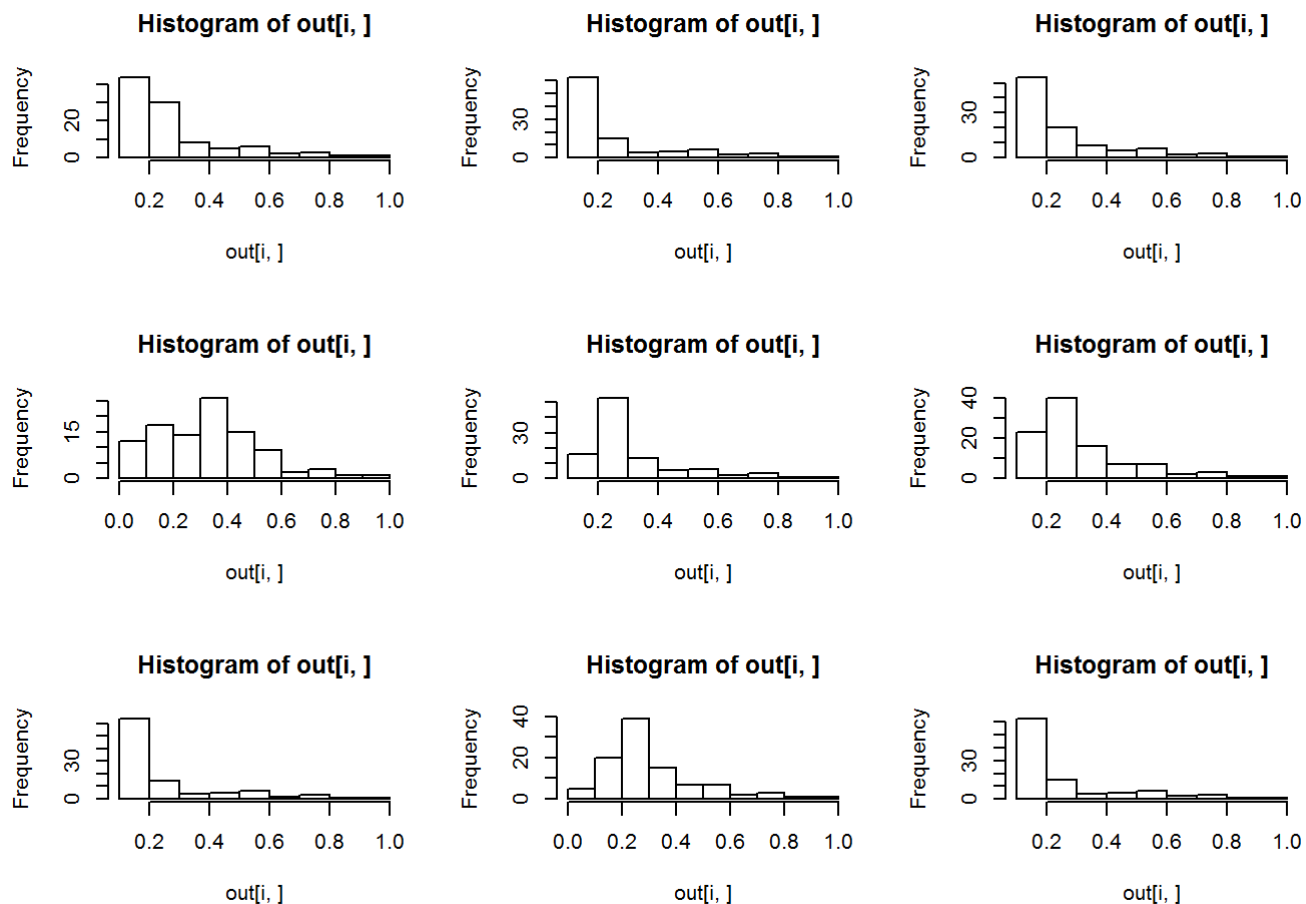
```
## [1] 0.2091492 0.3757193 0.4151315
```

```
## [1] 0.3278744 0.0503129 0.6218127
```

```
## [1] 0.05948781 0.70659750 0.23391469
```

```
## [1] 0.1558262 0.5710377 0.2731361
```

```
## [1] 0.71791920 0.26089505 0.02118575
```

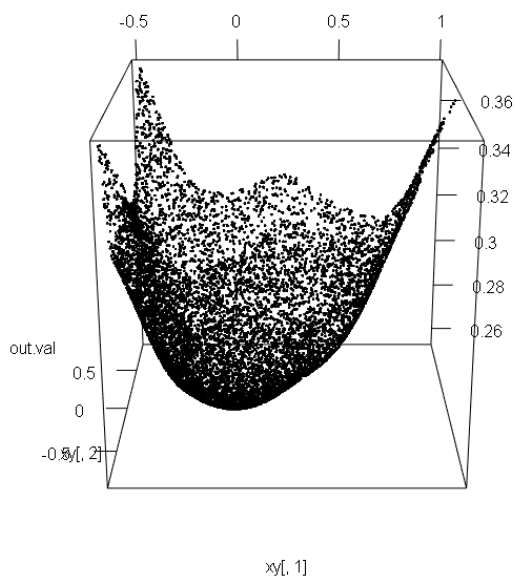


```
## [1] 0.3191097 0.4618257 0.2190646
```

```
par(mfcol=c(1,1))
```

失敗しない確率の期待値で選ぶなら

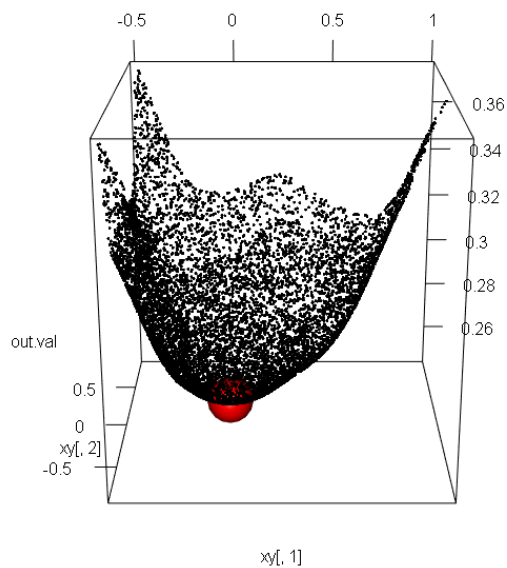
```
out.val <- apply(out,1,mean)
plot3d(xy[,1],xy[,2],out.val)
```



```
plot3d(xy[, 1], xy[, 2], out.val)
min.pt <- which(out.val==min(out.val))
print(Vs[min.pt,])
```

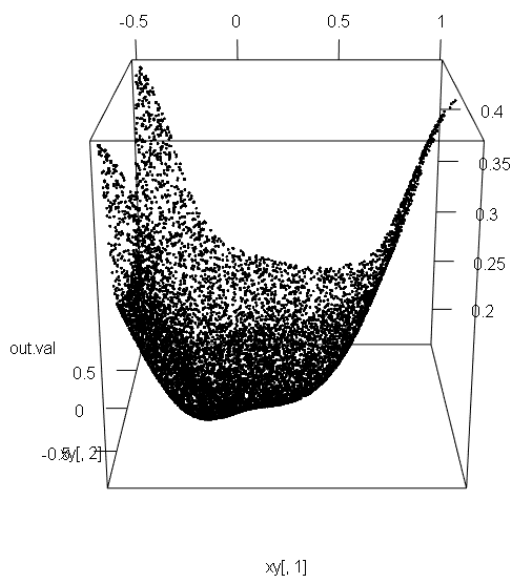
```
## [1] 0.3279570 0.3346456 0.3373974
```

```
spheres3d(c(xy[min.pt, ], out.val[min.pt]), col=2, radius=0.1)
```



失敗しない確率の中央値で選ぶなら

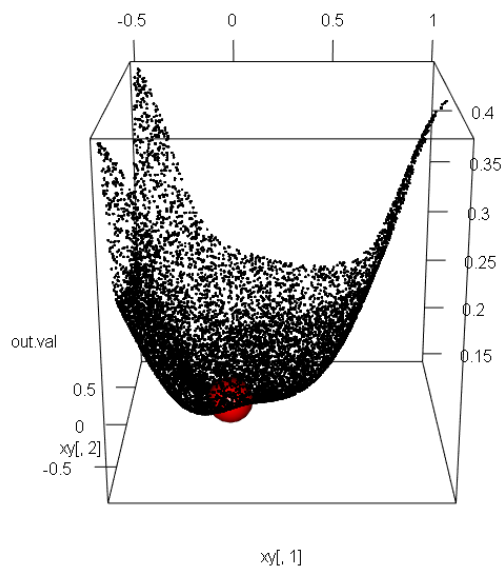
```
out.val <- apply(out, 1, quantile, probs=0.5)
plot3d(xy[, 1], xy[, 2], out.val)
```



```
plot3d(xy[, 1], xy[, 2], out.val)
min.pt <- which(out.val==min(out.val))
print(Vs[min.pt,])
```

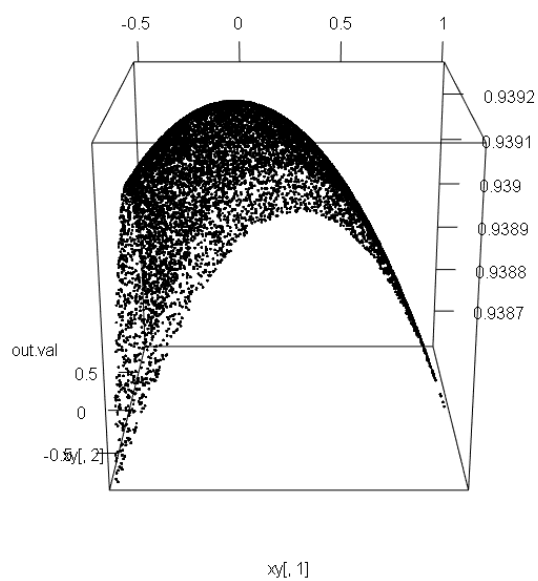
```
## [1] 0.3279570 0.3346456 0.3373974
```

```
spheres3d(c(xy[min.pt, ], out.val[min.pt]), col=2, radius=0.1)
```



失敗しない確率の最小値で選ぶなら

```
out.val <- apply(out, 1, max)
plot3d(xy[, 1], xy[, 2], out.val)
```



```
plot3d(xy[, 1], xy[, 2], out.val)
min.pt <- which(out.val==min(out.val))
print(Vs[min.pt,])
```

```
## [1] 0.9923491531 0.0005914427 0.0070594042
```

```
spheres3d(c(xy[min.pt, ], out.val[min.pt]), col=2, radius=0.1)
```

