水産資源データの特徴と解析

岡村　寛

この時間の目的

- 生態学・水産資源データの特徴を理解する
- Rによる簡単なデータの取り扱い（より高度な話は市野川さん）
- 線形回帰など簡単な分析法を使えるようになる
- 最小二乗法や最尤法，AICの使い方を学ぶ
- 一般化線形モデルの基礎について学ぶ（よりアドバンスな解説は西嶋さん）
- ランダム効果モデルの基礎
- シミュレーションによる精度計算や信頼区間について学ぶ（より詳しい話は秋田さん）

オープンソース・フリーの統計解析ソフトウェア

重要・最新の統計分析手法を網羅

モデル

優れたグラフィックス機能

視覚化

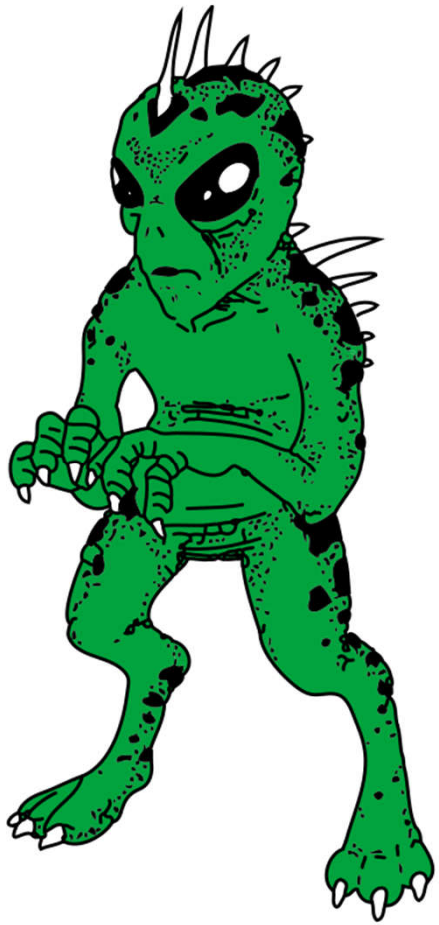
様々なソフトと連携して，計算やプレゼンの効率化が可能

コミュニケーション

水産資源（個体群）データ

dat1 <- read.csv(“dat1.csv”)

```
> dat1[sample(100,5),]
   count year site plant
16     9    1    5  tree
32     1    3    1 shrub
14     6    1    3  tree
72     2    7    1 shrub
35     3    3    4  tree
```

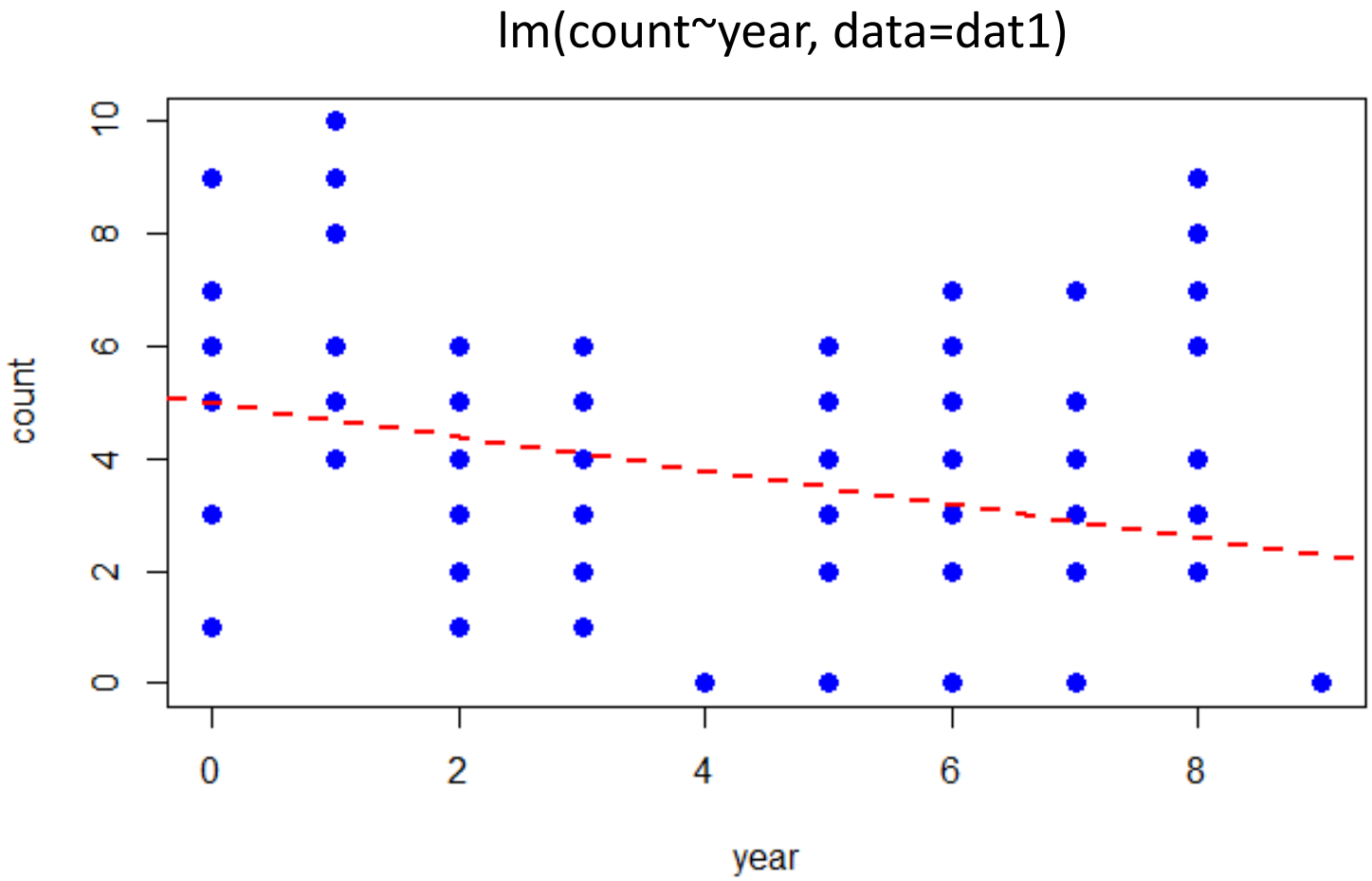


データの事前調査

```
table(dat1$count)
plot(count~year, data=dat1)
tapply(dat1$count,dat1$year,mean)

tapply(dat1$count,list(dat1$plant,dat1$year), mean)
plot(count~year, data=subset(dat1,plant=="tree"))
plot(count~year, data=subset(dat1,plant=="shrub"))
```

線形回帰モデル

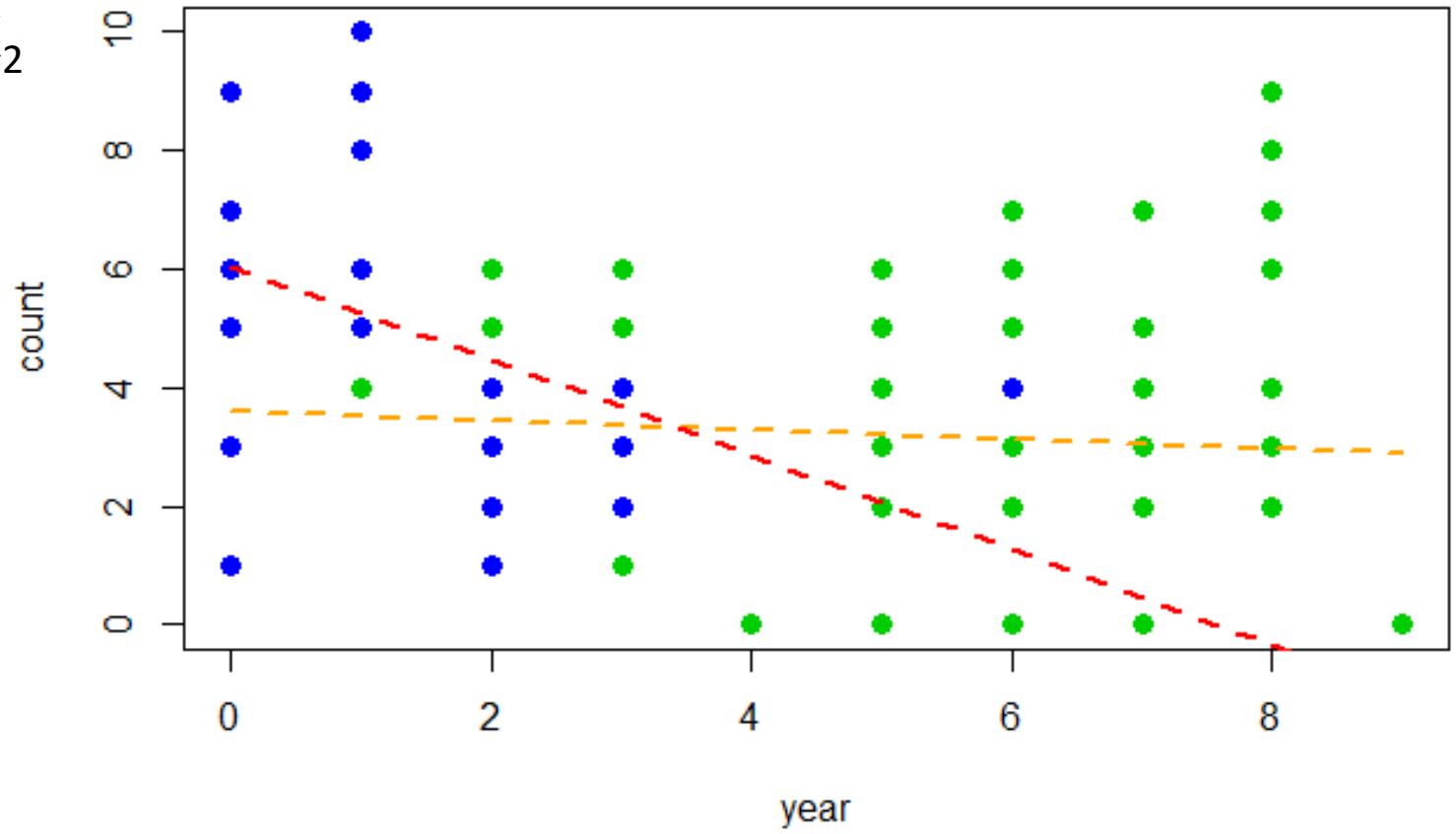
- Y = a + bX



重回帰モデル

- $Y=a+bX_1+cX_2$

lm(count~year*plant, data=dat1)



最小二乗法と最尤法

- 最小二乗法
- $\text{argmin } \Sigma[Y_i - (a+bX_i)]2$

- 最尤法
- $\text{argmax } \log L(\theta|X)$

確率モデルが正規分布なら同じになる

KL情報量と最尤法

- KL情報量：確率分布間の距離

真の確率分布Q(x)とモデルP(x|θ)の距離

$KL = \Sigma Q(x) \log [Q(x)/P(x|\theta)] = \Sigma\, Q(x) [\log Q(x) - \log P(x|\theta)]$

を最小にするためには,

$\Sigma Q(x) \log P(x|\theta) \approx (1/n)\Sigma \log P(x_i|\theta)$

対数尤度関数

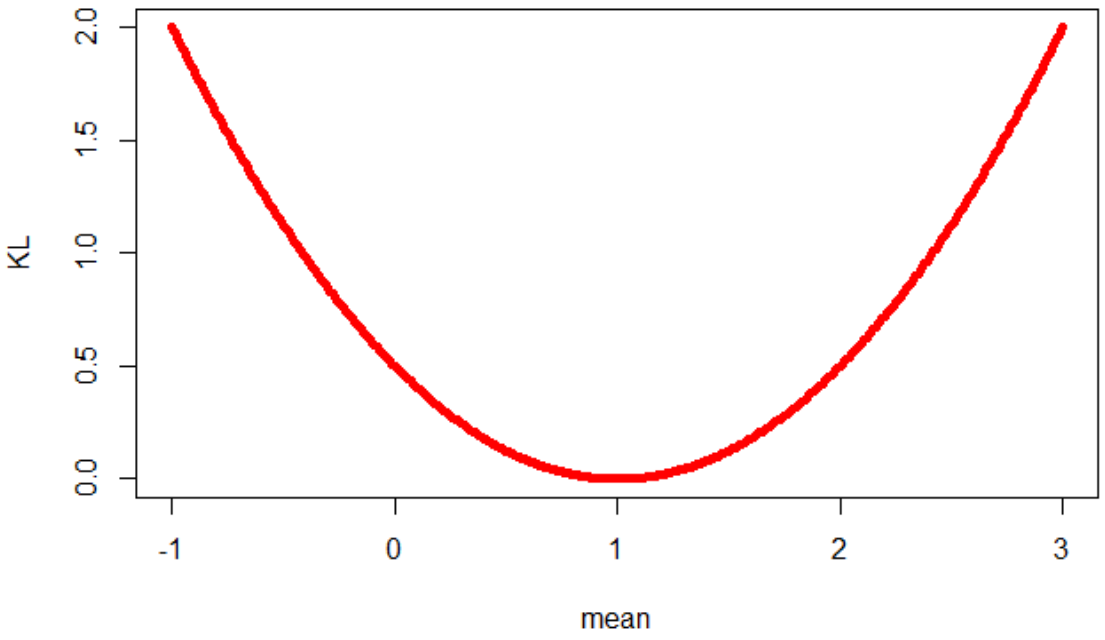
を最大化すれば良い.

対数尤度を最大にする推定量を最尤推定量という.

KL情報量

真がN(1,1)だとして，モデルN(m,1)のKL距離を計算

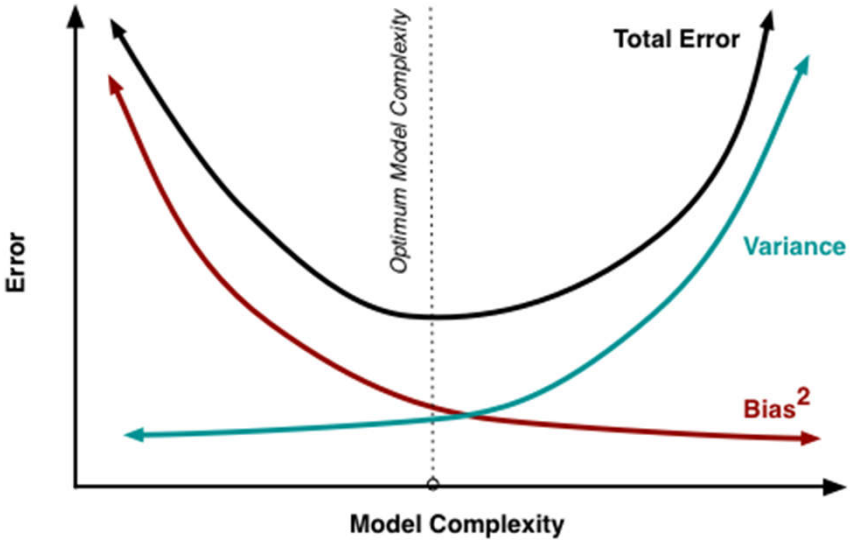
```
KL <- function(m)
integrate(function(x)
dnorm(x,1,1)*(dnorm(x,1,1,log=TR
UE)-dnorm(x,m,1,log=TRUE)),-
Inf,Inf)$value

KL <- Vectorize(KL,"m")

mu <- seq(-1,3,by=0.01)

plot(mu,KL(mu),xlab="mean",ylab
="KL",type="l",lwd=5,col="red")
```



AIC

• 実は，対数尤度関数はバイアスを持つ

$\Sigma Q(x) \log P(x|\theta) \approx (1/n)\{ \Sigma \log P(x_i|\theta) - p\}$



AIC = -2 LL + 2p

AICが小さいほど良いモデル

パラメータが多すぎるモデルはAICは大きくなる

パラメータが少なすぎるとLLが小さくAICは大きくなる

複雑でバイアス小/分散大と単純でバイアス大/分散小のバランスをとる

AIC

- 実際にバイアスがパラメータ数×2になるかどうかを確認する

```
Res.aic <- NULL
Sim <- 10000
set.seed(123)
for (n in c(10,100,500,1000)){
  z <- matrix(rnorm(n*Sim, 0, 1),nrow=Sim,ncol=n)
  TL <- apply(z,1,function(z) integrate(function(x) dnorm(x,0,1)*dnorm(x,mean(z),sqrt(var(z)*9/10),log=TRUE),-Inf,Inf)$value)
  EL <- apply(z,1,function(z) mean(dnorm(z,mean(z),sqrt(var(z)*9/10),log=TRUE)))
  Res.aic <- cbind(Res.aic, n*(EL-TL))
}
colnames(Res.aic) <- c(10,100,500,1000)
colMeans(Res.aic)
```

10	100	500	1000
2.848957	2.302538	2.000679	2.118348

教訓：原理を理解するのにプログラムを書いてみる
実感できる（百聞は一見に如かず）

AICによるモデル選択

```
res.n1 <- lm(count~year*plant,data=dat1)
> library(MuMIn)
> options(na.action = "na.fail") # 欠測値の取り扱いに関する設定（これをしないとdredgeが動かない）
> dredge(res.n1,rank="AIC")
Model selection table
  (Int) pln      yer pln:yer df   logLik   AIC delta weight
8 3.590   + -0.07941       +  5 -237.388 484.8  0.00  0.623
3 4.982     -0.29820          3 -240.437 486.9  2.10  0.218
4 4.521   + -0.23680          4 -240.190 488.4  3.60  0.103
2 3.121   +                  3 -241.886 489.8  5.00  0.051
1 3.640                      2 -245.303 494.6  9.83  0.005
```

一般化線形モデル

- 応答変数が0, 1, …. （カウントデータ）だったり，0/1（成功失敗）だったり

応答変数	範囲	確率分布
連続	-∞～+∞	正規分布
連続	0 ~ +∞	対数正規分布，ガンマ分布
離散	0,1,2,…	ポアソン分布，負の二項分布
離散	0, 1	二項分布

ポアソン回帰モデル

- optimを使って計算してみよう！

```
Poisson.reg <- function(p,dat){
  lambda <- exp(p[1]+p[2]*dat$year)
  -sum(dat$count*log(lambda)-lambda)
}

optim(c(log(6),-0.07),Poisson.reg,dat=dat1,method="BFGS")
```

ポアソン回帰モデル

glm関数

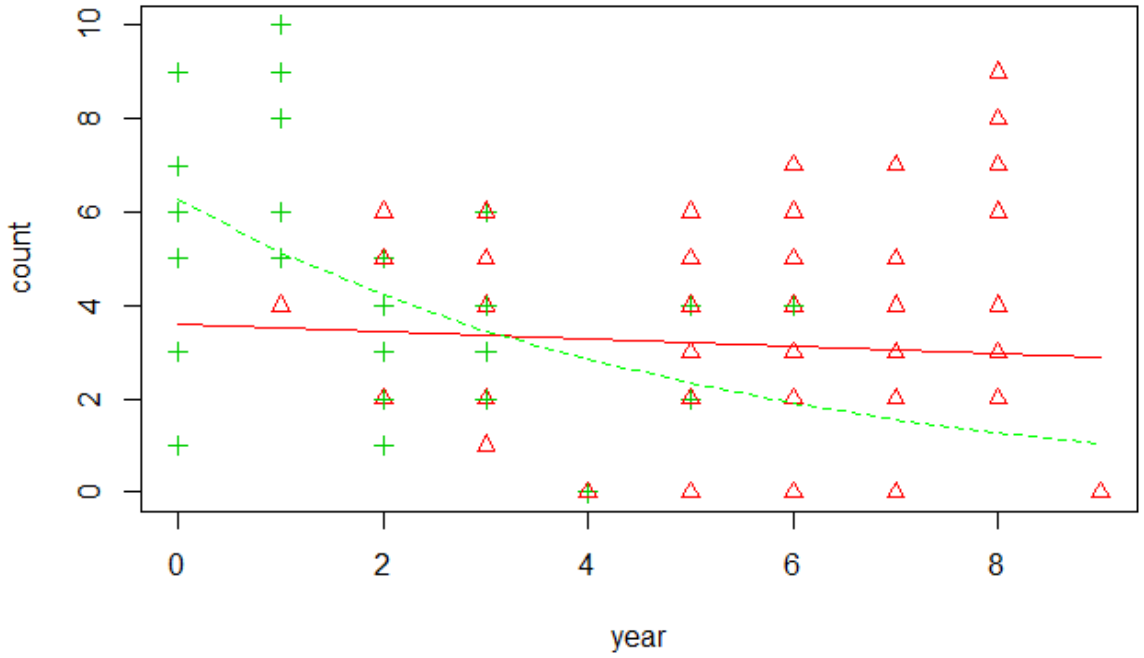
glm(count~year,family=poisson,data=dat1)

glm(count~year*plant,family=poisson,data=dat1)

ポアソン回帰モデル

モデル選択

```
res.p1 <- glm(count~year*plant,family=poisson,data=dat1)
dredge(res.p1,rank="AIC")
```



ロジスティック回帰モデル

- optimを使って計算しよう！

```
Logit.reg <- function(p,dat){
  prob <- 1/(1+exp(-(p[1]+p[2]*dat$year)))
  -sum(dat$count*log(prob)+(10-dat$count)*log(1-prob))
}

optim(c(0,0),Logit.reg,dat=dat1,method="BFGS")
```

ロジスティック回帰モデル

```
glm(cbind(count,10-count)~year,family=binomial,data=dat1)

res.l1 <- glm(cbind(count,10-
count)~year*plant,family=binomial,data=dat1)
dredge(res.l1,rank="AIC")
```

N混合モデル

- 複数の調査地で繰り返しサンプリングを行う

調査地の平均個体数 $Po(N|\lambda)$

調査地の発見数 $Bi(x|n, 1 – (1 – r)^N)$

Nは観測されない潜在変数（ランダム効果）であるので，すべての可能性に対して足し合わせた周辺尤度を最大化してやる

N混合モデル

発見数モデル

```
detect.f <- function(p, dat, N, n=10){
  y <- dat$count
  X <- cbind(1,as.numeric(dat$plant)-
1)
  r <- c(1/(1+exp(-X%*%p)))

  dbinom(y,n,1-(1-r)^N)
}
```

個体数モデル

```
pop.f <- function(p, dat, N){
  X <- cbind(1,dat$year)
  lambda <- c(exp(X%*%p))

  dpois(N,lambda)
}
```

N混合モデル

```
popdet.f <- function(p, dat, max.N=100, n=10){
  Pop <- function(i) {
    pop1 <- pop.f(p[3:4],dat[i,],0:max.N)
    pop1 <- pop1/sum(pop1)
    pop1
  }
  like <- sapply(1:nrow(dat), function(i) sum(detect.f(p[1:2],dat[i,],0:max.N,n)*Pop(i)))
  -sum(log(like))
}

(res.nm <- optim(c(0,0,0,0),popdet.f,dat=dat1,max.N=100,method="BFGS"))
```

精度計算と信頼区間

- デルタ法

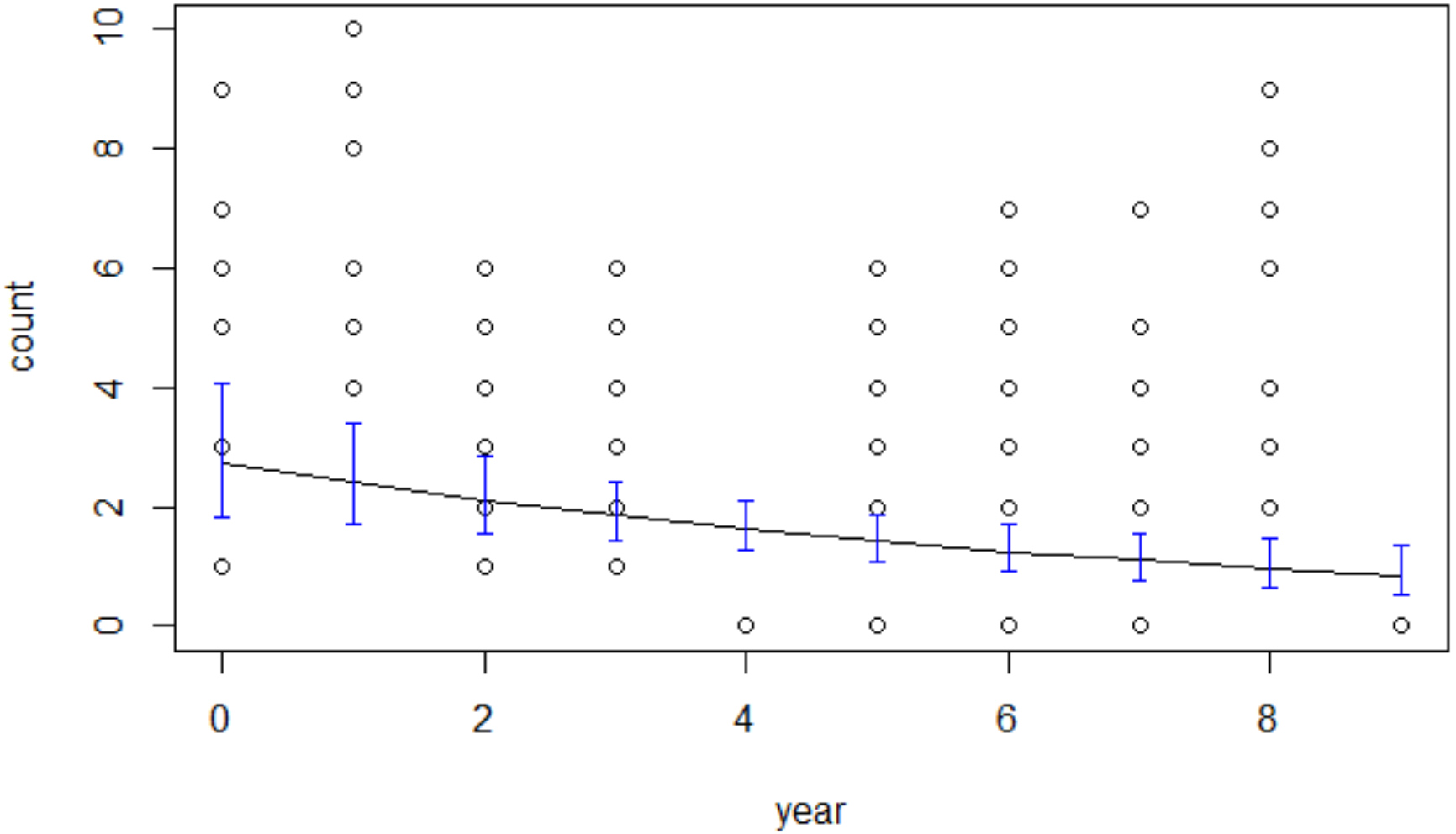
$\mathrm{var}[f(x)] = (df/dx)^2 \, \mathrm{var}(x)$

- 対数正規信頼区間

$CV(x) = SE(x)/E(x)$

$[x/C,\ xC]$

$C = \exp(z\,[\log(1+CV(x)^2)]^{1/2})$


count
0
2
4
6
8
10
0
2
4
6
8
year

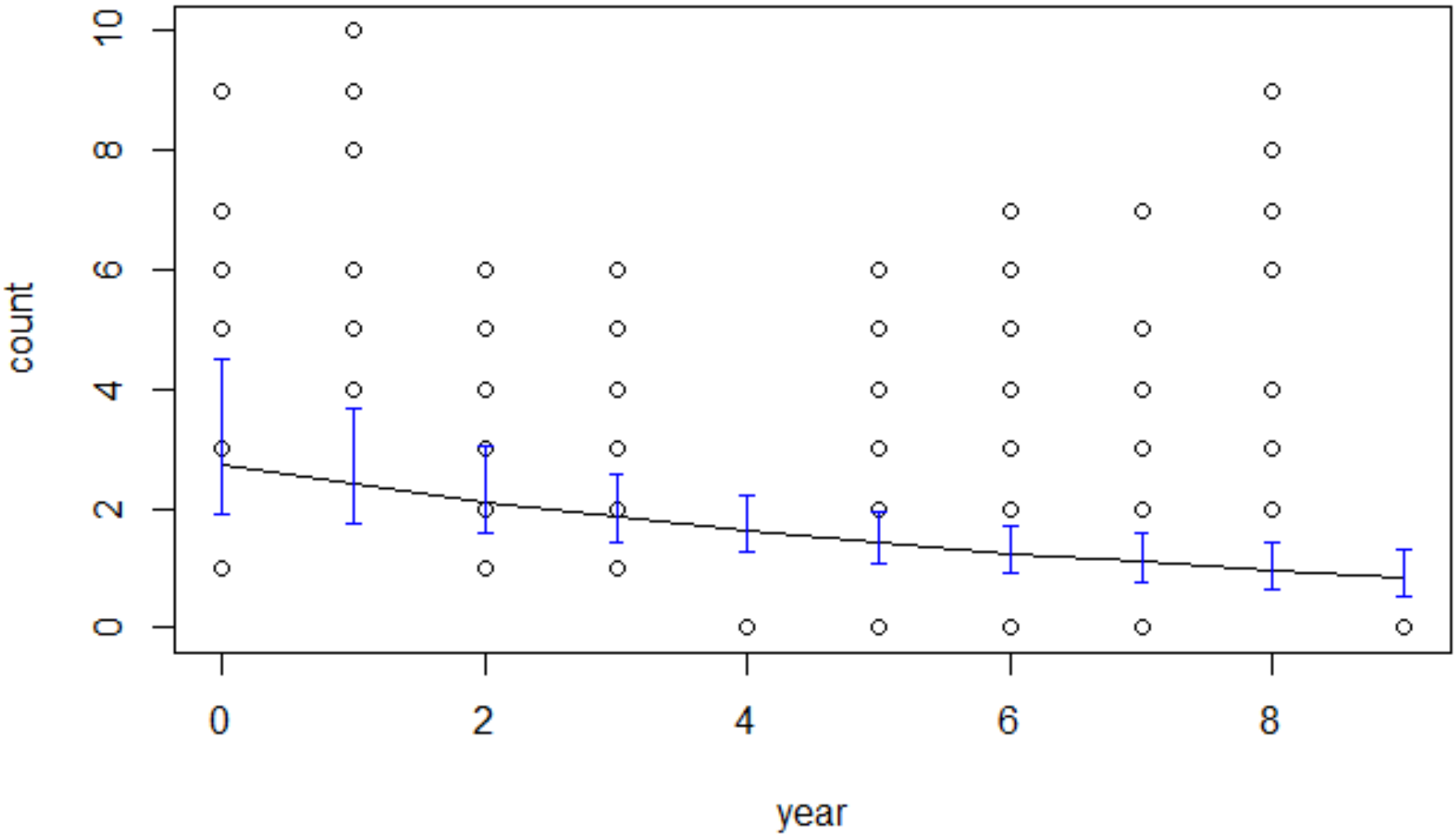
ブートストラップ法

- データを真の確率分布の近似と考えてやる
- それからランダムサンプリング（リサンプリング）すれば，現実にあり得た繰り返しランダムサンプルを再現できる（現実にはサンプルは1個しかないのに）
- リサンプリングしたデータそれぞれに対して統計量を計算すれば，任意の統計量の確率分布を推定できる！

ブートストラップ法の種類	内容
ノンパラメトリックブートストラップ	データをリサンプリングして統計量の計算を繰り返す
パラメトリックブートストラップ	データをにフィットした確率分布からデータ生成を繰り返す
残差ブートストラップ	データと予測モデルの残差をリサンプリングする

ブートストラップ法

```
B <- 1000
res.p0 <- glm(count~year,family=poisson,data=dat1)
pred.p0 <- predict(res.p0,type="response")
b.pb <- NULL
for (i in 1:B){
  count.pb <- rpois(100,pred.p0)
  res.p0.pb <- glm(count.pb~year,family=poisson,data=dat1)
  b.pb <- c(b.pb, res.p0.pb$coef[2])
}
quantile(b.pb,probs=c(0.025,0.975))
```


count
0
2
4
6
8
10
0
2
4
6
8
year

TMBによるN混合モデル計算

- optimの計算は時間がかかる
- 現代資源解析では，シミュレーションを多用する必要がある
- コンピュータ能力の向上によって計算のスピードは問題なくなった，と言われるが，データの量・計算の複雑さは爆発的に増加 → 計算しきれない → 高速計算プログラムの価値はますます高い
- TMB（Template Model Builder）　超速い

TMBによるN混合モデル計算

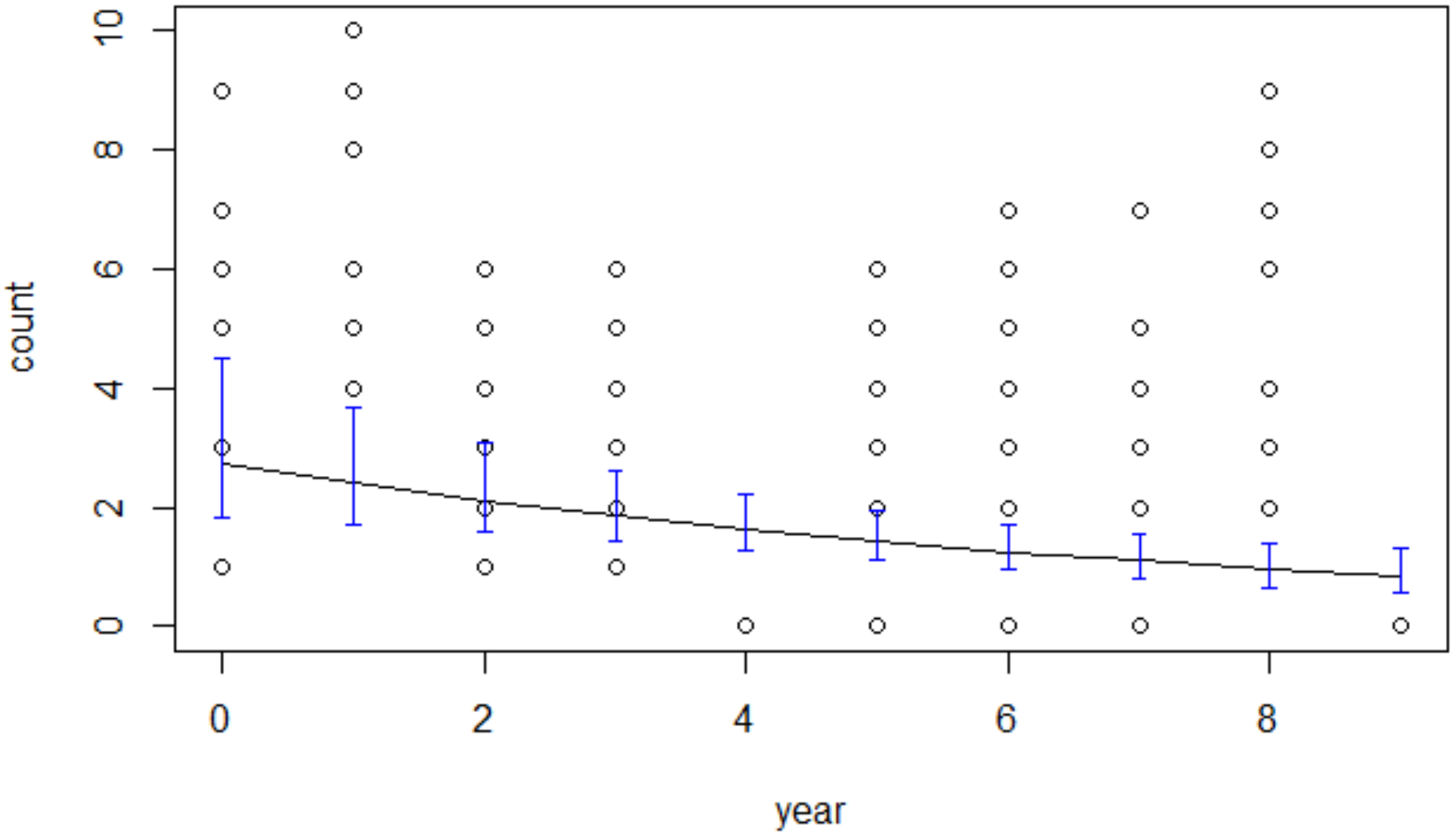
- 自動微分
- ラプラス近似
- ベイズ推定のためStanと連携
- C++でプログラムを書かないといけないが，書いてしまえばストレスフリーになる
- TMB+INLA → VAST（Throson + 西嶋さん）

TMBによるN混合モデル計算

```
library(TMB)
compile("nm.cpp")
dyn.load(dynlib("nm"))
```

TMBによるN混合モデル計算

```
dat2 <- dat1[,-3]
dat2[,3] <- as.numeric(dat2[,3])-1
dat <- list(Nmax=100, n=10, DAT=as.matrix(dat2))
parms <- list(P=c(0,0,0,0))
obj <- MakeADFun(data=dat,parameters=parms,DLL="nm")
(res.nm.tmb <-
optim(obj$par,obj$fn,obj$gr,method="BFGS",hessian=TRUE))
```


count
0
2
4
6
8
10
0
2
4
6
8
year

まとめ

水産資源学のデータ解析の基本である

- 最尤法，AIC
- 回帰，一般化線形モデル
- 信頼区間
- ブートストラップ法
- Template Model Builder

について学んだ