Re-revisiting dependent data

Eli Gurarie

StatR 301 - Lecture 7
University of Washington - Seattle

May 16, 2013

Dependent data

Already in this sequence:

- We visited - briefly - **time-series** - in the last lecture of StatR 101

Dependent data

Already in this sequence:

- We visited - briefly - **time-series** - in the last lecture of StatR 101
- Assaf (in his ultra-advanced) way - managed to slip-in sideways **Markov Chains** when talking about MCMC for Bayesian Methods (Week 2) ....
- ... and an estimation of an **auto-correlated, moving-average** structure in Week 4 (Generalized Estimating Equations and Mixed-Effects Models).

Dependent data

Already in this sequence:

- We visited - briefly - **time-series** - in the last lecture of StatR 101
- Assaf (in his ultra-advanced) way - managed to slip-in sideways **Markov Chains** when talking about MCMC for Bayesian Methods (Week 2) ....
- ... and an estimation of an **auto-correlated, moving-average** structure in Week 4 (Generalized Estimating Equations and Mixed-Effects Models).

Goals this week and next, is to linger a bit more on these topics: to think about ways in which correlations influence inference, but ways in which dependencies themselves are worth exploring, to introduce **stochastic modeling** in general.

Along the way (esp. in lab), we'll play with some interesting large datasets, and push our graphical visualization skills a bit, e.g. *3-D*, *animation*, *interactive graphics*, *mapping*.

Linear models

Simple:

$$\begin{aligned} Y_i &= \mathbf{X}\beta + \epsilon_i \\ \epsilon &\sim \mathcal{N}(0, \sigma^2) \end{aligned}$$

where **X** is the *linear predictor* (or "model matrix"), β are the parameters, and ϵ_i are i.i.d.

Generalized:

$$\begin{aligned} \mathrm{Y}_i &\sim \mathit{Distribution}(\mu) \\ \mu &= \mathrm{E}\left(Y_i\right) = g^{-1}(\mathbf{X}\beta) \end{aligned}$$

where *Distribution* is exponential family and $g(\mu)$ is the *link function*.

Note: In both cases i is *unordered*: $Y_1, Y_2, Y_3 \ldots Y_n$ can be reshuffled: $Y_5, Y_3, Y_2, Y_n \ldots Y_3$, with no effect on the results (as long as the covariates are also reshuffled), and $\mathrm{Cov}\left(()\, Y_i, Y_j\right) = \mathrm{E}\left(Y_i Y_j\right) - \mathrm{E}\left(Y_i\right)\mathrm{E}\left(Y_j\right) = 0$

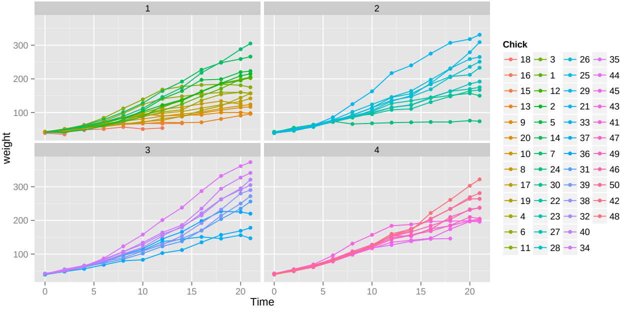
You've already seen one kind of dependency:

Mixed-effect:

$$g(\mathrm{E}(Y_i)) = \mathbf{X}\beta + Z\gamma$$

where Z is a $n \times m$ matrix of 1 and 0, indicating the group of the n'th observation.

```r
qplot(Time, weight, data = ChickWeight, col = Chick, facets = ~Diet) + geom_line() +
    guides(col = guide_legend(ncol = 4))
```



Here - the dependency is within the grouping of the individual, which is modelled as a "random effect" ... but basically this is like having just another structural index.

Some theoretical background: Variance-Covariance Structures

Linear regession ("ordinary least squares"):

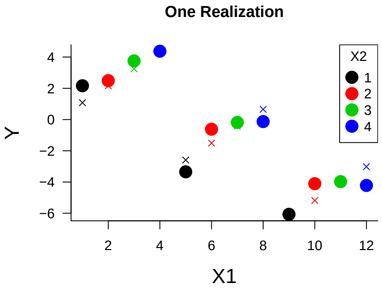
$$y = \mathbf{X}\beta + \epsilon$$

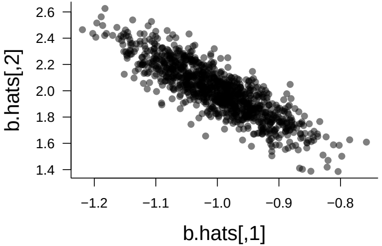
y is $n \times 1$ response, $\mathbf{X}$ is n model matrix; β is $p \times 1$ vector of parameters, ϵ is $n \times 1$ vector of errors. Assuming $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_n)$ (where $\mathbf{I}_n$ is $n \times n$ identity matrix) gives **OLS** estimator of β:

$$\begin{aligned} \hat{\beta} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'y \\ V(\hat{\beta}) &= \sigma^2(\mathbf{X}'\mathbf{X})^{-1} \end{aligned}$$

To convince yourself OLS works

```
myols <- function(sigma = 1, beta = c(-1,2))
{
  X <- cbind(1:12, rep(1:4,3))
  y <- X %*% beta + rnorm(12, 0, sigma)
b.hat <- solve(t(X)%*% X) %*% t(X) %*% y
  list(X=X, y=y, b.hat=b.hat)
}
ols <- myols(); ols$b.hat

##          [,1]
## [1,] -0.9182
## [2,]  1.9999

b.hats <- matrix(0, nrow=1000, ncol=2)
for(i in 1:1000) b.hats[i,] <- myols()$b.hat[,1]
var(b.hats)

##           [,1]     [,2]
## [1,]  0.005561 -0.01306
## [2,] -0.013062  0.04210

(sigma <- 1)*solve(t(ols$X)%*% ols$X)

##          [,1]     [,2]
## [1,]  0.00625 -0.01458
## [2,] -0.01458  0.04514
```





Some theoretical background: Variance-Covariance Structures

Assume more generally that $\epsilon \sim \mathcal{N}(0, \Sigma)$ has nonzero off-diagonal entries correspondiong to *correlated errors*. If Σ is known, the log-likelihood for the model is maximized with:

$$\hat{\beta}_{gls} = (\mathbf{X}\Sigma^{-1}\mathbf{X})^{-1}\mathbf{X}\Sigma^{-1}\mathbf{y}$$

For example, when Σ is a diagonal matrix of *unequal* error variances (heteroskedasticity), then $\hat{\beta}$ is the weighted-least-squares (WLS) estimator.

In a real application, of course, Σ is not known, and must be estimated along with the regression coefficients β ... *But there are way too many elements in* Σ - *(this many:* $n(n+1)/2$*)*.

Some theoretical background: Variance-Covariance Structures

Assume more generally that $\epsilon \sim \mathcal{N}(0, \Sigma)$ has nonzero off-diagonal entries correspondiong to *correlated errors*. If Σ is known, the log-likelihood for the model is maximized with:

$$\hat{\beta}_{gls} = (\mathbf{X}\Sigma^{-1}\mathbf{X})^{-1}\mathbf{X}\Sigma^{-1}\mathbf{y}$$

For example, when Σ is a diagonal matrix of *unequal* error variances (heteroskedasticity), then $\hat{\beta}$ is the weighted-least-squares (WLS) estimator.

In a real application, of course, Σ is not known, and must be estimated along with the regression coefficients β ... *But there are way too many elements in* Σ - *(this many: $n(n+1)/2$)*.

A large part of dealing with dependent data is identifying a *tractable, relatively simple* form for that residual variance-covariance matrix, and then fitting/choosing models that take it into account.

Recall Time-Series models

Time-series:

- Any data (or realization of random process) that are indexed by *time*
- $i \rightarrow t$: $Y_1, Y_2, Y_3 \ldots Y_n$ becomes $Y_{t_1}, Y_{t_2}, Y_{t_3} \ldots Y_{t_n}$
- **Discrete-time time-series**:
  - Data collected at regular (Annual, Monthly, Daily, Hourly, etc.) intervals.
  - t usually denoted (and ordered) $1, 2, 3, \ldots n$.
- **Continuous-time time-series**:
  - Data collected at arbitrary intervals: $t_i \in \{T_{min}, T_{max}\}$.

The autoregressive model

- First order autoregressive model: AR(1)

$$X_t = \phi_1 X_{t-1} + Z_t$$

where $Z_t \sim \mathcal{N}(0, \sigma^2)$ is **White Noise**.

The autoregressive model

- First order autoregressive model: AR(1)

$$X_t = \phi_1 X_{t-1} + Z_t$$

where $Z_t \sim \mathcal{N}(0, \sigma^2)$ is **White Noise**.

- Second order autoregressive: AR(2)

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + Z_t$$

The autoregressive model

- First order autoregressive model: AR(1)

$$X_t = \phi_1 X_{t-1} + Z_t$$

where $Z_t \sim \mathcal{N}(0, \sigma^2)$ is **White Noise**.

- Second order autoregressive: AR(2)

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + Z_t$$

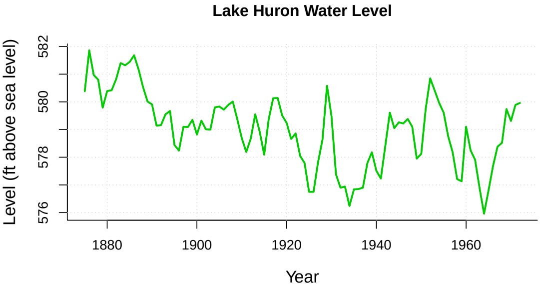
- p-th order autoregressive model: AR(p)

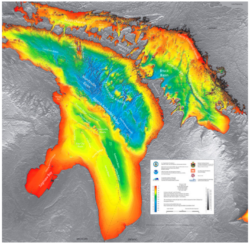
$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \ldots + \phi_p Z_{t-p} + Z_t$$

- Note: these assume $\mathrm{E}(X) = 0$. Relaxing this gives (for AR(1)):

$$X_t = \phi_1 (X_{t-1} - \mu) + Z_t + \mu$$

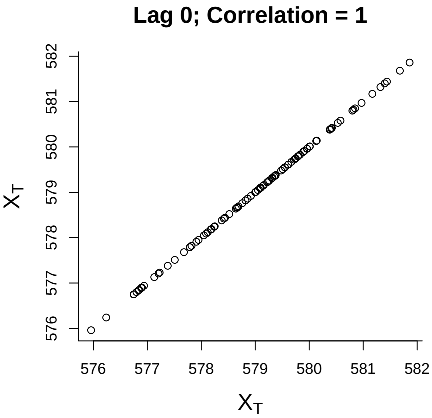
Concept 1: Autocorrelation

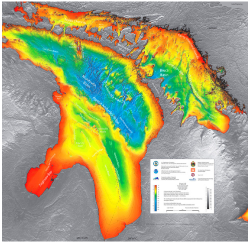




Autocorrelation function (acf): $\rho(h) = \text{Cor}\left(X_t, X_{t+h}\right)$

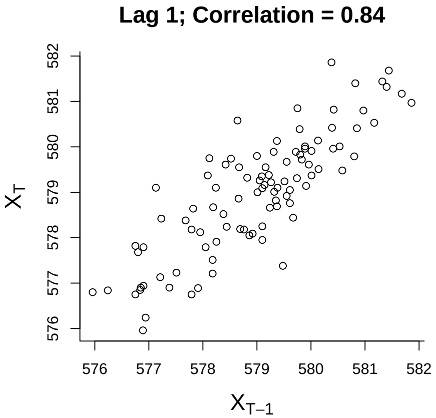
Concept 1: Autocorrelation

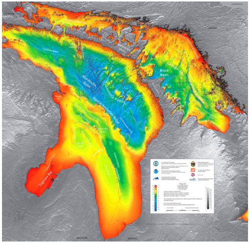




Autocorrelation function (acf): $\rho(h) = \text{Cor}\,(X_t, X_{t+h})$

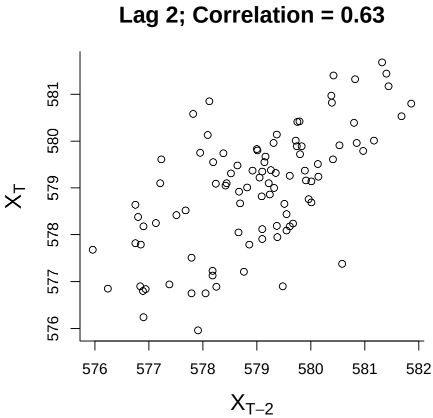
Concept 1: Autocorrelation

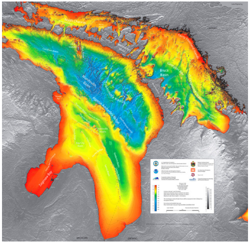




Autocorrelation function (acf): $\rho(h) = \mathrm{Cor}\left(X_t, X_{t+h}\right)$

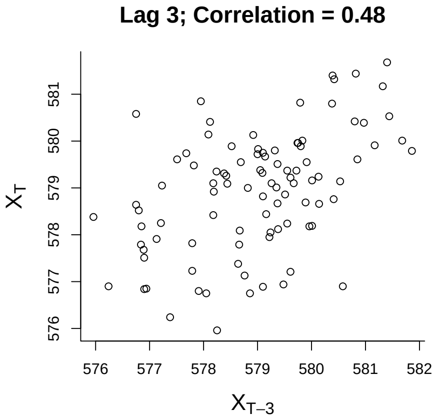
Concept 1: Autocorrelation

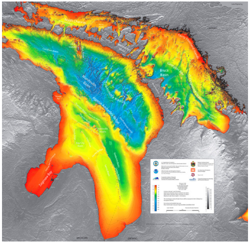




Autocorrelation function (acf): $\rho(h) = \mathrm{Cor}\left(X_t, X_{t+h}\right)$

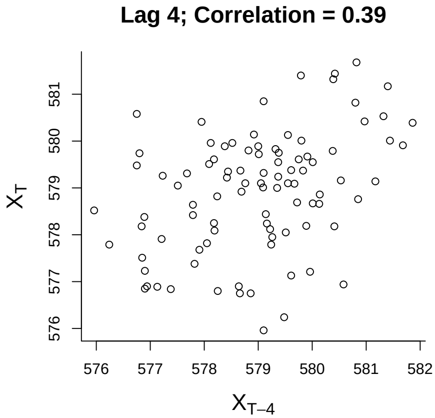
Concept 1: Autocorrelation

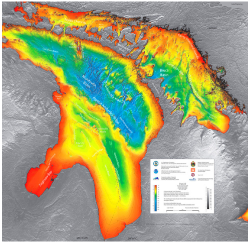




Autocorrelation function (acf): $\rho(h) = \text{Cor}\left(X_t, X_{t+h}\right)$

Concept 1: Autocorrelation



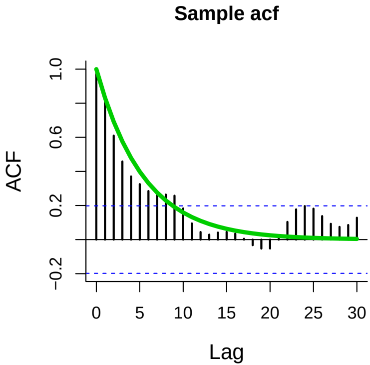


Autocorrelation function (acf): $\rho(h) = \mathrm{Cor}\,(X_t, X_{t+h})$

AR(1) Theoretical prediction

For the AR(1) model, the theoretical acf is:

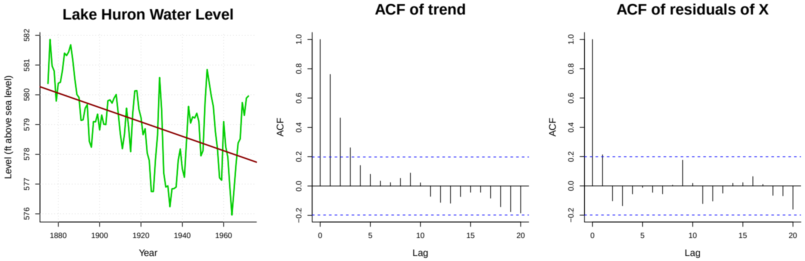
$$\rho(h) = \phi_1^h$$



If the sample acf looks exponential - probably an AR(1) model.

Fitting a linear trend and correlation to Lake Huron

$$
\begin{aligned}
Y_t &= \beta_0 + \beta_1 t + \epsilon_t \\
\epsilon_t &= \phi_1 \epsilon_{t-1} + Z_t
\end{aligned}
$$



- $\widehat{\beta}_0 = 625$ ft; $\widehat{\beta}_1 = -0.024$ ft/year; $\widehat{\phi}_1 = 0.79$; $\widehat{\sigma}^2 = 0.513$ ft^2

Fitting a linear trend and correlation to Lake Huron in R

Version 1: Two steps

```
LH.trend <- lm(LakeHuron ~ time(LakeHuron))
eps <- residuals(LH.trend)
eps.lm <- lm(eps[-1] ~ eps[-length(eps)])
summary(LH.trend)$coef

##                  Estimate Std. Error t value  Pr(>|t|)
## (Intercept)     625.5549   7.764293  80.568 5.809e-90
## time(LakeHuron)  -0.0242   0.004036  -5.996 3.545e-08

summary(eps.lm)$coef

##                    Estimate Std. Error t value  Pr(>|t|)
## (Intercept)         0.01529    0.07272  0.2102 8.340e-01
## eps[-length(eps)]   0.79112    0.06590 12.0044 9.502e-21
```

Version 2: One Step (using generalized least squares)

```
require(nlme)
LH.gls <- gls(LakeHuron ~ time(LakeHuron), correlation = corAR1(form = ~1))
summary(LH.gls)

## Generalized least squares fit by REML
##   Model: LakeHuron ~ time(LakeHuron)
##   Data: NULL
##     AIC   BIC logLik
##   225.8 236.1 -108.9
##
## Correlation Structure: AR(1)
##  Formula: ~1
##  Parameter estimate(s):
##    Phi
## 0.8248
##
## Coefficients:
##                 Value Std.Error t-value p-value
## (Intercept)     616.5    24.363  25.305  0.0000
## time(LakeHuron)   0.0     0.013  -1.535  0.1282
##
##  Correlation:
##                 (Intr)
## time(LakeHuron) -1
##
## Standardized residuals:
##      Min       Q1      Med       Q3      Max
## -2.11193 -0.57664 -0.05772  0.53782  1.82271
##
## Residual standard error: 1.261
## Degrees of freedom: 98 total; 96 residual
```

Note the residual variance-covariance for an AR(1) is:

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & \phi & \phi^2 & \phi^3 & \dots & \phi^n \\ \phi & 1 & \phi & \phi^2 & \dots & \phi^{n-1} \\ \phi^2 & \phi & 1 & \phi & \dots & \phi^{n-2} \\ \phi^3 & \phi^2 & \phi & 1 & & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots \\ \phi^n & \phi^{n-1} & \phi^{n-2} & \dots & \dots & 1 \end{bmatrix}$$

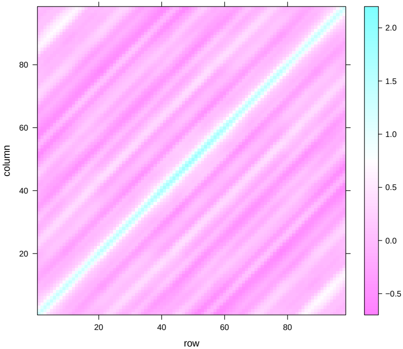
So Σ is estimated with only two parameters, ϕ and σ^2

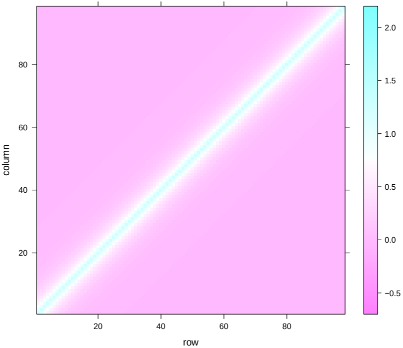
Empirical Σ matrix

```
res <- lm(LakeHuron~time(LakeHuron))$res
n <- length(res); V <- matrix(NA, nrow=n, ncol=n)
for(i in 1:n) V[n-i+1,(i:n - i)+1] <- res[i:n]
ind <- upper.tri(V); V[ind] <- t(V)[ind]
require(lattice)
levelplot(var(V, na.rm=TRUE), at=seq(-.7,2.2,length=100))
```

Theoretical Σ matrix

```
sigma.hat <-  LH.gls$sigma
phi.hat <- 0.8
require(lattice)
V.hat <- outer(1:n, 1:n,
              function(x,y) phi.hat^abs(x-y))
levelplot(sigma.hat*V.hat, at=seq(-.7,2.2,length=100))
```





Passes the "eye-test" ... but can we trust the "eye-test"?

How to determine the order of an AR(p) model?

The base `ar()` function, which uses AIC:

```
ar(LakeHuron)

##
## Call:
## ar(x = LakeHuron)
##
## Coefficients:
##      1       2
##  1.054  -0.267
##
## Order selected 2  sigma^2 estimated as  0.508

LH.lm <- lm(LakeHuron ~ time(LakeHuron))
ar(LH.lm$res)

##
## Call:
## ar(x = LH.lm$res)
##
## Coefficients:
##      1       2
##  0.971  -0.275
##
## Order selected 2  sigma^2 estimated as  0.501
```

Strong suggestion that there is a negative second-order correlation.

Fitting an AR(2) model

```
require(nlme)
LH.gls2 <- gls(LakeHuron ~ time(LakeHuron), correlation = corARMA(p = 2))
summary(LH.gls2)
```

```
## Generalized least squares fit by REML
##   Model: LakeHuron ~ time(LakeHuron)
##   Data: NULL
##   AIC   BIC logLik
##   221 233.8 -105.5
##
## Correlation Structure: ARMA(2,0)
##  Formula: ~1
##  Parameter estimate(s):
##    Phi1    Phi2
##  1.0203 -0.2741
##
## Coefficients:
##                 Value Std.Error t-value p-value
## (Intercept)     619.6    17.491   35.43  0.0000
## time(LakeHuron)   0.0     0.009   -2.32  0.0223
##
##  Correlation:
##                 (Intr)
## time(LakeHuron) -1
##
## Standardized residuals:
##      Min       Q1      Med       Q3      Max
## -2.16625 -0.57960  0.01071  0.61705  2.03976
##
## Residual standard error: 1.186
## Degrees of freedom: 98 total; 96 residual
```

Fitting an AR(2) model

```r
require(nlme)
LH.gls2 <- gls(LakeHuron ~ time(LakeHuron), correlation = corARMA(p = 2))
summary(LH.gls2)
```

```
## Generalized least squares fit by REML
##   Model: LakeHuron ~ time(LakeHuron)
##   Data: NULL
##   AIC   BIC logLik
##   221 233.8 -105.5
##
## Correlation Structure: ARMA(2,0)
##  Formula: ~1
##  Parameter estimate(s):
##    Phi1    Phi2
##  1.0203 -0.2741
##
## Coefficients:
##                 Value Std.Error t-value p-value
## (Intercept)     619.6    17.491   35.43  0.0000
## time(LakeHuron)   0.0     0.009   -2.32  0.0223
##
##  Correlation:
##                 (Intr)
## time(LakeHuron) -1
##
## Standardized residuals:
##       Min        Q1       Med        Q3       Max
## -2.16625 -0.57960  0.01071  0.61705  2.03976
##
## Residual standard error: 1.186
## Degrees of freedom: 98 total; 96 residual
```

So .... the temporal regression *IS* significant!?

Comparing models ...

```r
LH.gls0.1 <- gls(LakeHuron ~ 1, correlation = corAR1(form = ~1), method = "ML")
LH.gls0.2 <- gls(LakeHuron ~ 1, correlation = corARMA(p = 2), method = "ML")
LH.gls1.1 <- gls(LakeHuron ~ time(LakeHuron), correlation = corAR1(form = ~1),
    method = "ML")
LH.gls1.2 <- gls(LakeHuron ~ time(LakeHuron), correlation = corARMA(p = 2),
    method = "ML")

anova(LH.gls0.1, LH.gls0.2, LH.gls1.1, LH.gls1.2)
```

```
##           Model df   AIC   BIC logLik   Test L.Ratio p-value
## LH.gls0.1     1  3 219.2 226.9 -106.6
## LH.gls0.2     2  4 215.3 225.6 -103.6 1 vs 2   5.930  0.0149
## LH.gls1.1     3  4 218.4 228.8 -105.2
## LH.gls1.2     4  5 212.4 225.3 -101.2 3 vs 4   8.054  0.0045
```

Comparing models ...

```
LH.gls0.1 <- gls(LakeHuron ~ 1, correlation = corAR1(form = ~1), method = "ML")
LH.gls0.2 <- gls(LakeHuron ~ 1, correlation = corARMA(p = 2), method = "ML")
LH.gls1.1 <- gls(LakeHuron ~ time(LakeHuron), correlation = corAR1(form = ~1),
    method = "ML")
LH.gls1.2 <- gls(LakeHuron ~ time(LakeHuron), correlation = corARMA(p = 2),
    method = "ML")

anova(LH.gls0.1, LH.gls0.2, LH.gls1.1, LH.gls1.2)

##           Model df   AIC   BIC logLik   Test L.Ratio p-value
## LH.gls0.1     1  3 219.2 226.9 -106.6
## LH.gls0.2     2  4 215.3 225.6 -103.6 1 vs 2   5.930  0.0149
## LH.gls1.1     3  4 218.4 228.8 -105.2
## LH.gls1.2     4  5 212.4 225.3 -101.2 3 vs 4   8.054  0.0045
```

Newest conclusions

- The water level in Lake Huron IS dropping, and there is a high first order and significant negative second-order auto-correlation to the water level.
- Even for very simple time-series and questions about simple linear trends ... it is easy to make false inference (both see patterns that are not there, or fail to detect patterns that are there) if you don't take correlations into account!

Yet another important twist: Moving average models

- First-order moving average model: MA(1)

$$_t = Z_t + \psi Z_{t-1}$$

  i.e. the "random shock" (aka "innovation") depends on the previous "random shock".

- q-th order moving average model: MA(q)

$$\epsilon_t = \psi_1 Z_{t-1} + \psi_2 Z_{t-2} + ... + \psi_q Z_{t-q} + Z_t$$

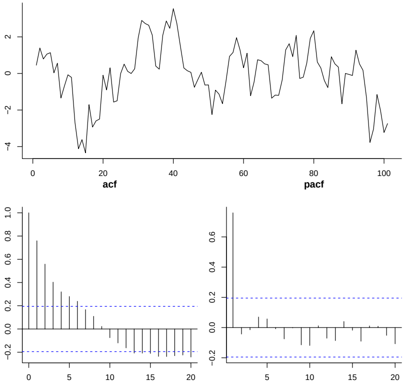
Assessed via the **partial autocorrelation function**

$$\begin{aligned}\phi(k) \quad = \quad & \mathrm{Corr}(X_t - \mathcal{P}(X_t \mid X_{t+1}, \ldots, X_{t+k-1}), \\ & X_{t+k} - \mathcal{P}(X_{t+k} \mid X_{t+1}, \ldots, X_{t+k-1}))\end{aligned} \tag{1}$$

where: $\mathcal{P}(Y \mid X)$ is the best linear projection of Y on X, i.e. we've regressed away all of the lags and are left only with the cross-correlation of the residuals.

PACF(h) of an AR(1) is 0 at $h > 0$

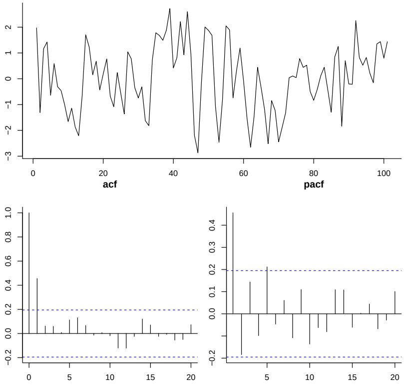
```r
phi <- 0.8; nu <- rnorm(100)
esp.AR <- c(rnorm(1),rep(NA,100))
for(i in 1:100)
  esp.AR[i+1] <- phi*esp.AR[i] + nu[i]
```




```r
acf(esp.AR, main = "acf")
pacf(esp.AR, main = "pacf")
```

ACF(h) of an MA(1) is 0 at $h > 1$

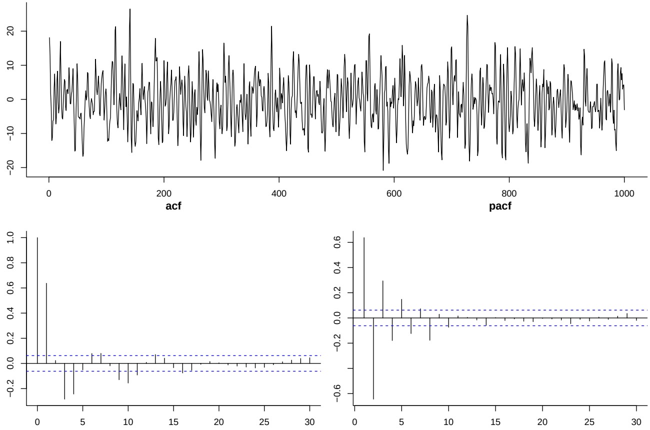
```r
psi <- 0.8;  nu <- rnorm(101)
esp.MA <- c(rnorm(1),rep(NA,100))
for(i in 2:101)
 esp.MA[i] <- nu[i] + psi*nu[i-1]
```




```r
acf(esp.MA, main = "acf")
pacf(esp.MA, main = "pacf")
```

ARMA(p,q) model

```
Y <- arima.sim(model = list(ar = c(0.9, -.5), ma = c(-2,-4)), 1000)
```



Selecting ARMA(p,q)

```
get.aic.table <- function(Y) {
    aic.table <- AIC(arima(Y, c(1, 0, 0)), arima(Y, c(0, 0, 1)), arima(Y, c(1,
        0, 1)), arima(Y, c(2, 0, 0)), arima(Y, c(0, 0, 2)), arima(Y, c(1, 0,
        2)), arima(Y, c(2, 0, 1)), arima(Y, c(2, 0, 2)), arima(Y, c(3, 0, 2)),
        arima(Y, c(3, 0, 3)))
    aic.table[order(aic.table$AIC), ]
}
```

```
Y <- arima.sim(model =
  list(ar = c(0.9, -.5),
      ma = c(-2,-4)), 1000)
get.aic.table(Y)

##                        df  AIC
## arima(Y, c(2, 0, 2))  6 5590
## arima(Y, c(3, 0, 2))  7 5591
## arima(Y, c(3, 0, 3))  8 5592
## arima(Y, c(2, 0, 1))  5 5595
## arima(Y, c(1, 0, 2))  5 5640
## arima(Y, c(0, 0, 2))  4 5642
## arima(Y, c(1, 0, 1))  4 5721
## arima(Y, c(2, 0, 0))  4 5874
## arima(Y, c(0, 0, 1))  3 5915
## arima(Y, c(1, 0, 0))  3 6389
```

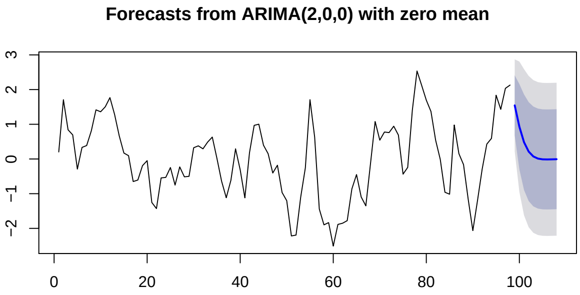
```
get.aic.table(LakeHuron)

##                        df   AIC
## arima(Y, c(1, 0, 1))  4 214.5
## arima(Y, c(2, 0, 0))  4 215.3
## arima(Y, c(1, 0, 2))  5 216.5
## arima(Y, c(2, 0, 1))  5 216.5
## arima(Y, c(2, 0, 2))  6 218.4
## arima(Y, c(1, 0, 0))  3 219.2
## arima(Y, c(3, 0, 2))  7 219.7
## arima(Y, c(3, 0, 3))  8 221.2
## arima(Y, c(0, 0, 2))  4 230.9
## arima(Y, c(0, 0, 1))  3 255.3
```

```
res <- LH.lm$res
get.aic.table(res)

## Warning:  possible convergence
problem:  optim gave code = 1

##                        df   AIC
## arima(Y, c(2, 0, 0))  4 210.5
## arima(Y, c(1, 0, 1))  4 210.5
## arima(Y, c(1, 0, 2))  5 212.1
## arima(Y, c(2, 0, 1))  5 212.2
## arima(Y, c(2, 0, 2))  6 214.1
## arima(Y, c(3, 0, 2))  7 215.2
## arima(Y, c(1, 0, 0))  3 216.6
## arima(Y, c(3, 0, 3))  8 216.7
## arima(Y, c(0, 0, 2))  4 217.8
## arima(Y, c(0, 0, 1))  3 235.2
```

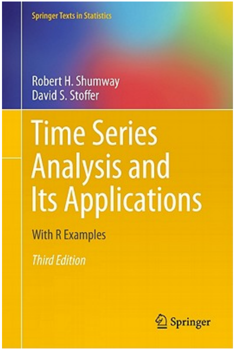
(I couldn't find a more "automated" way to do this ... which is strange)

Forecasting

```
require(forecast)
(LH.forecast <- forecast(Arima(LH.lm$res, order = c(2, 0, 0), include.mean = FALSE)))

##     Point Forecast    Lo 80 Hi 80   Lo 95 Hi 95
##  99       1.545024  0.6785 2.412  0.2198 2.870
## 100       0.929893 -0.2986 2.158 -0.9489 2.809
## 101       0.482674 -0.8942 1.860 -1.6231 2.588
## 102       0.213123 -1.2126 1.639 -1.9674 2.394
## 103       0.073021 -1.3654 1.511 -2.1268 2.273
## 104       0.011054 -1.4297 1.452 -2.1924 2.215
## 105      -0.010247 -1.4513 1.431 -2.2141 2.194
## 106      -0.013532 -1.4546 1.428 -2.2174 2.190
## 107      -0.010603 -1.4517 1.430 -2.2145 2.193
## 108      -0.006698 -1.4478 1.434 -2.2106 2.197

plot(LH.forecast)
```



Some more resources



Code, comments, datasets, examples: http://www.stat.pitt.edu/stoffer/tsa3/

Rob Hyndman's list of time series resources in R

Basic facilities

stats Contains substantial time series capabilities including the ts class for regularly spaced time series. Also ARIMA modelling, structural models, time series plots, acf and pacf graphs, classical decomposition and STL decomposition.

Forecasting and univariate modelling

forecast Lots of univariate time series methods including automatic ARIMA modelling, exponential smoothing via state space models, and the forecast class for consistent handling of time series forecasts. Part of the **forecasting** bundle.

tseries GARCH models and unit root tests.

FitAR Subset AR model fitting

partsm Periodic autoregressive time series models

pear Periodic autoregressive time series models

Forecasting and univariate modelling

ltsa Methods for linear time series analysis

dlm Bayesian analysis of Dynamic Linear Models.

timsac Time series analysis and control

fArma ARMA Modelling

fGarch ARCH/GARCH modelling

BootPR Bias-corrected forecasting and bootstrap prediction intervals for autoregressive time series

gsarima Generalized SARIMA time series simulation

bayesGARCH Bayesian Estimation of the GARCH(1,1) Model with t innovations

Resampling and simulation

boot Bootstrapping, including the block bootstrap with several variants.

meboot Maximum Entropy Bootstrap for Time Series

Rob Hyndman's list of time series resources in R

Decomposition and filtering

robfilter Robust time series filters
mFilter Miscellaneous time series filters useful for smoothing and extracting trend and cyclical components.
ArDec Autoregressive decomposition
wmtsa Wavelet methods for time series analysis based on Percival and Walden (2000)
wavelets Computing wavelet filters, wavelet transforms and multiresolution analyses
signalextraction Real-time signal extraction (direct filter approach)
bspec Bayesian inference on the discrete power spectrum of time series

Unit roots and cointegration

tseries Unit root tests and methods for computational finance.
urca Unit root and cointegration tests
uroot Unit root tests including methods for seasonal time series

Nonlinear time series analysis

nlts R functions for (non)linear time series analysis
tseriesChaos Nonlinear time series analysis
RTisean Algorithms for time series analysis from nonlinear dynamical systems theory.
tsDyn Time series analysis based on dynamical systems theory
BAYSTAR Bayesian analysis of threshold autoregressive models
fNonlinear Nonlinear and Chaotic Time Series Modelling
bentcableAR Bent-Cable autoregression

Dynamic regression models

dynlm Dynamic linear models and time series regression
dyn Time series regression
tpr Regression models with time-varying coefficients.

Rob Hyndman's list of time series resources in R

Multivariate time series models

mAr Multivariate AutoRegressive analysis

vars VAR and VEC models

MSBVAR Markov-Switching Bayesian Vector Autoregression Models

tsfa Time series factor analysis

dse Dynamic system equations including multivariate ARMA and state space models.

brainwaver Wavelet analysis of multivariate time series

Functional data

far Modelling Functional AutoRegressive processes

Continuous time data

cts Continuous time autoregressive models

sde Simulation and inference for stochastic differential equations.

Irregular time series

zoo Infrastructure for both regularly and irregularly spaced time series.

its Another implementation of irregular time series.

fCalendar Chronological and Calendarical Objects

fSeries Financial Time Series Objects

xts Provides for uniform handling of R's different time-based data classes

Rob Hyndman's list of time series resources in R

Time series and forecasting in R — Time series packages on CRAN 68

Time series data

fma Data from Makridakis, Wheelwright and Hyndman (1998) *Forecasting: methods and applications*. Part of the **forecasting** bundle.

expsmooth Data from Hyndman, Koehler, Ord and Snyder (2008) *Forecasting with exponential smoothing*. Part of the **forecasting** bundle.

Mcomp Data from the M-competition and M3-competition. Part of the **forecasting** bundle.

FinTS R companion to Tsay (2005) *Analysis of financial time series* containing data sets, functions and script files required to work some of the examples.

TSA R functions and datasets from Cryer and Chan (2008) *Time series analysis with applications in R*

TSdbi Common interface to time series databases

fame Interface for FAME time series databases

fEcofin Ecofin - Economic and Financial Data Sets

Time series and forecasting in R — Time series packages on CRAN 69

Miscellaneous

hydrosanity Graphical user interface for exploring hydrological time series

pastecs Regulation, decomposition and analysis of space-time series.

RSEIS Seismic time series analysis tools

paleoTS Modeling evolution in paleontological time-series

GeneTS Microarray Time Series and Network Analysis

fractal Fractal Time Series Modeling and Analysis

http://robjhyndman.com/research/

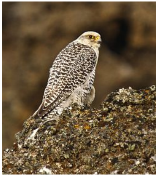
And now for something quite different ...

Or is it?

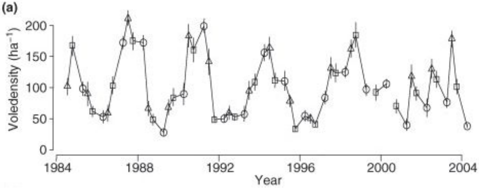
Population cycles

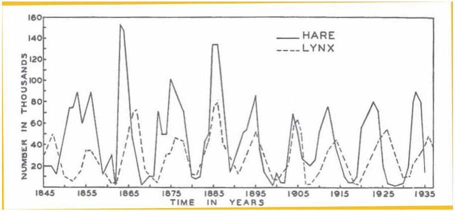
Voles

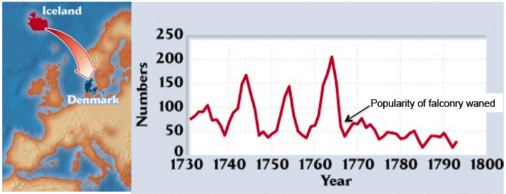
Lynx and Hare



Gyrfalcon





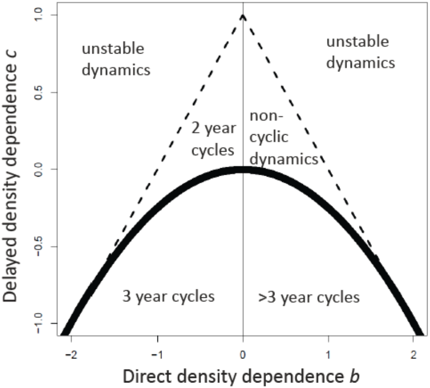


Why do cycles occur?

- One explanation: Cycles are the result of temporal lags in reponses of population *to their own density*.
  - i.e. **delayed density dependence**
- "Intrinsic" population control
  - Direct density dependence: inertia (i.e. positive autocorrelation) from recent high/low populations.
  - Delayed density dependence: high populations at some greater lags lead to local depletion or resources, or density dependent lower recruitment, which propagates with some delay through the popualtion.

Delayed density dependence

$$N_i(t+1) - N_i(t)$$
$$= a_{it} + \underbrace{b_{it}}_{\text{Direct density dependence}} N_i(t) + \underbrace{c_{it}}_{\text{Delayed density dependence}} N_i(t-1) + \varepsilon_{it}$$



Mathematical analysis of difference equations predicts different cyclic patterns (or lack thereof) depending of values of b and c.

Delayed density dependence

Note that the equation is essentially an AR(2) process!

$$\begin{aligned} N_{t-1} - N_t &= a + b\, N_t + c\, N_{t-1} + Z_t \\ N_{t-1} &= a + (1+b)\, N_t + c\, N_{t-1} + Z_t \end{aligned}$$

And easy to estimate with standard linear regression.

Note: Since these are counts, and many populations fluctuate over large orders of magnitude, and population processes (especially growth) tend to be exponential ... very often this analysis is done with $\log(N)$ rather than N.

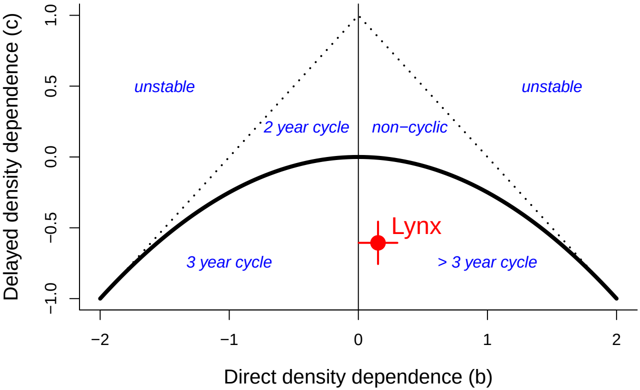
Lynx and Hare analysis

Estimating these parameters for the Canada Lynx gives (e.g. with `lm(lynx.lag0   lynx.lag1 + lynx.lag2)` or (more quickly) with `arma(lynx, order= c(2,0))` :

```
Coefficient(s):
ar1       ar2       intercept
1.1524    -0.6062   710.0962
```

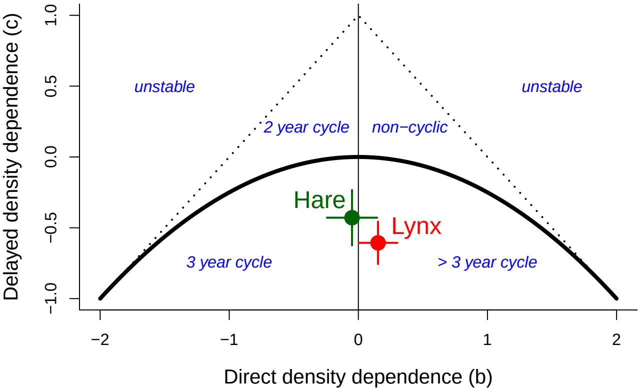
so: $\hat{b} = 0.1524$ and $\hat{c} = -0.6062$.

Lynx analysis



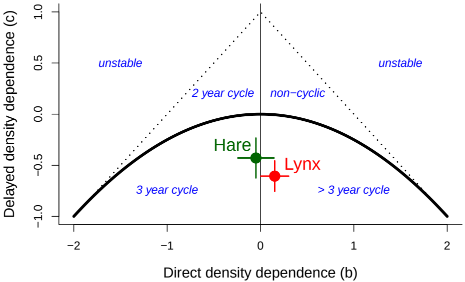
$$L_t - L_{t-1} = 710 + 0.15\, L_{t-1} - 0.606\, L_{t-2}$$

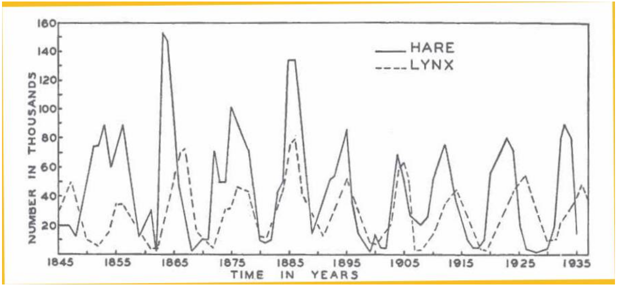
Lynx and Hare analysis



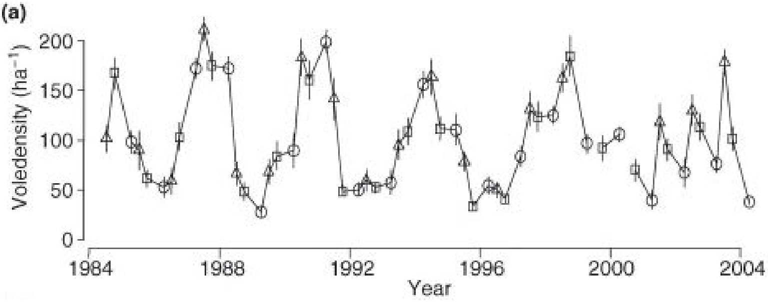
$$L_t - L_{t-1} = 710 + 0.15\, L_{t-1} - 0.606\, L_{t-2}$$
$$H_t - H_{t-1} = 21.6 - 0.05\, H_{t-1} - 0.43\, H_{t-2}$$

Compare to time series





Collapsing cyclicity



Opinion

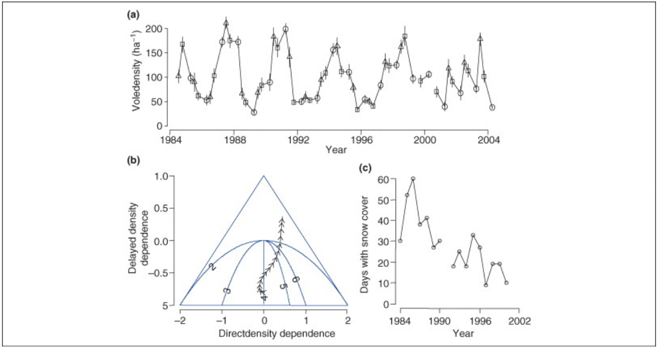
Trends in Ecology and Evolution Vol.23 No.2

Collapsing population cycles

Rolf A. Ims, John-André Henden and Siw T. Killengreen

Department of Biology, University of Tromsø, NO-9037 Tromsø, Norway

Collapsing cyclicity



- Some evidence that climate change (specifically: milder winters) is leading to a "collapse" in cyclicity.
- Basic tool: AR(2) models, but with additional covariates including winter and summer lengths.