Random effects in RTMB

Anders Nielsen, Ethan Lawler, & Sean Anderson

an@aqua.dtu.dk

Example: Paired observations

- Two methods A and B to measure blood cell count (to check for the use of doping).
- Paired study.

Person ID	Method A	Method B
1	5.5	5.4
2	4.4	4.9
3	4.6	4.5
4	5.4	4.9
5	7.6	7.2
6	5.9	5.5
7	6.1	6.1
8	7.8	7.5
9	6.7	6.3
10	4.7	4.2

- It must be expected that two measurements from the same person are correlated, so a paired t-test is the correct analysis
- The t-test gives a p-value of 5.1%, which is a borderline result...
- But more data is available

- In addition to the planned study 10 persons were measured with only one method
- Want to use all data, which is possible with random effects
- Assume these 20 are ramdomly selected from a population where the blod cell count is normally distributed
- Consider the following model:
  $C_i = \alpha(M_i) + B(P_i) + \varepsilon_i, \quad i = 1 \ldots 30$
  $\alpha(M_i)$ the 2 fixed method effects
  $B(P_i) \sim \mathcal{N}(0, \sigma_P^2)$ the 20 random effects
  $\varepsilon_i \sim \mathcal{N}(0, \sigma_R^2)$ measurement noise
  All $B(P_i)$ and ε_i are assumed independent
- This model uses all data and gives a 95% c. i. for the method bias $\alpha(A) - \alpha(B)$ which is: $(0.04; 0.41)$.
- Notice that now there is a (slightly) significant method bias.

Person ID	Method A	Method B
1	5.5	5.4
2	4.4	4.9
3	4.6	4.5
4	5.4	4.9
5	7.6	7.2
6	5.9	5.5
7	6.1	6.1
8	7.8	7.5
9	6.7	6.3
10	4.7	4.2
11		5.1
12		4.4
13		4.5
14		5.3
15		7.5
16	5.7	
17	6.0	
18	7.5	
19	6.5	
20	4.2	

Random effects

Random effect model

- In purely fixed effects models we have
  - Random variables we observe
  - Model parameters we want to estimate
- In random effects models we have
  - Random variables we observe
  - Random variables we do NOT observe
  - Model parameters we want to estimate
- This model class is very useful and goes by many names: random effects models, mixed models, latent variable models, state-space models, frailty models, hierarchical models, ...

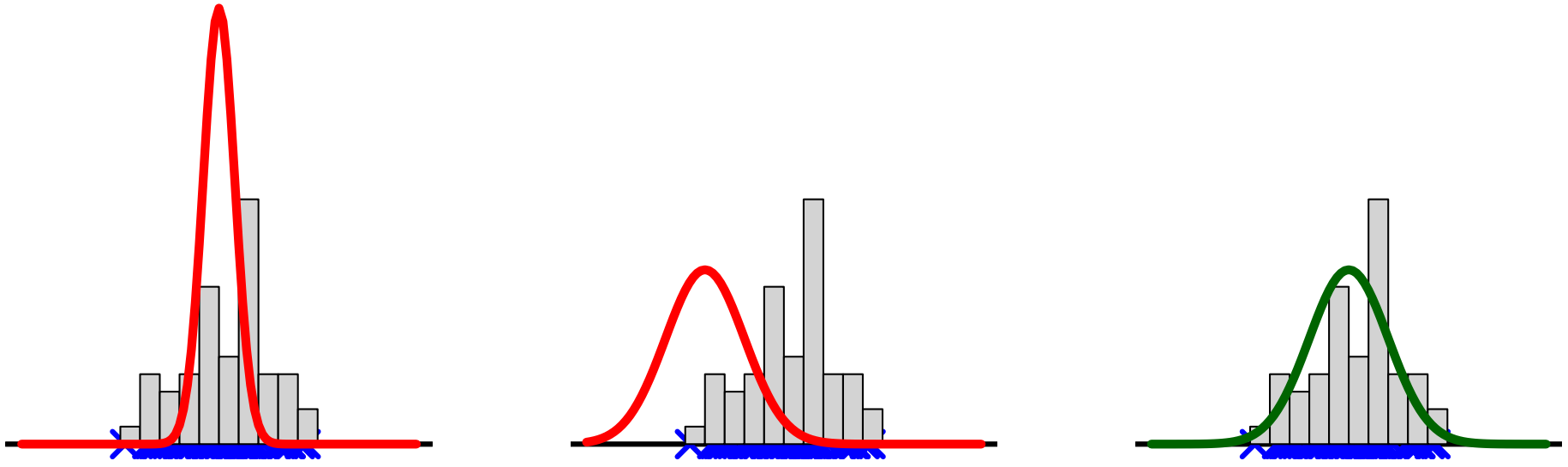
So the difference is that we have some quantity with a distribution, which is unobserved.

Reminder: Estimation with purely fixed effects

- We have:

  Observations: $y = (y_1, y_2, \ldots, y_n)$

  Parameters (μ, σ) in model: $y_i \sim N(\mu, \sigma^2)$



- Choose parameters which makes our model best match the data (optimize likelihood).

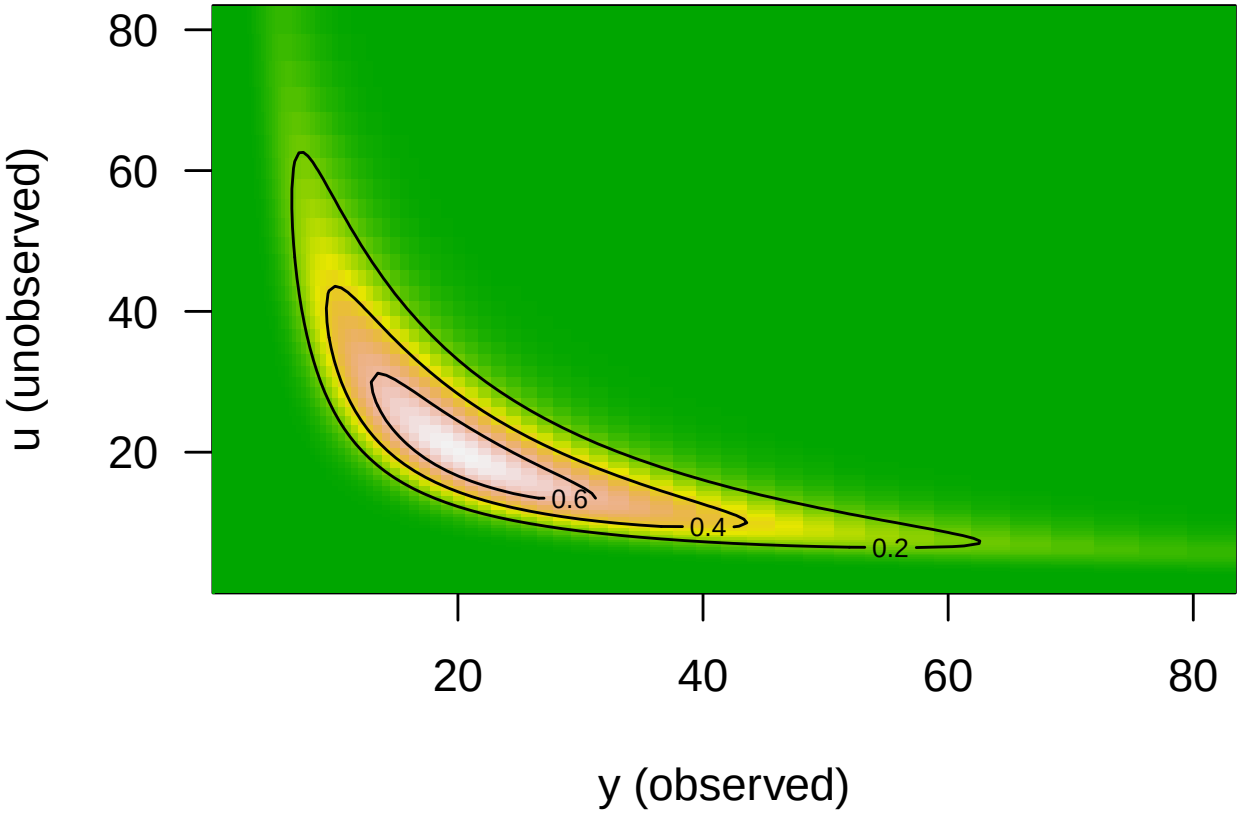
Estimation with random effects

- We have:

  Observations: y

  Unobserved random effects: u

  Parameters (θ) in model: $(y, u) \sim D(\theta)$



- How do we estimate our parameters when some of our observations are not observed?

Estimation with random effects — 2

- The banana is only an intermediate calculation

  1: Joint model (banana) is determined from model parameters θ

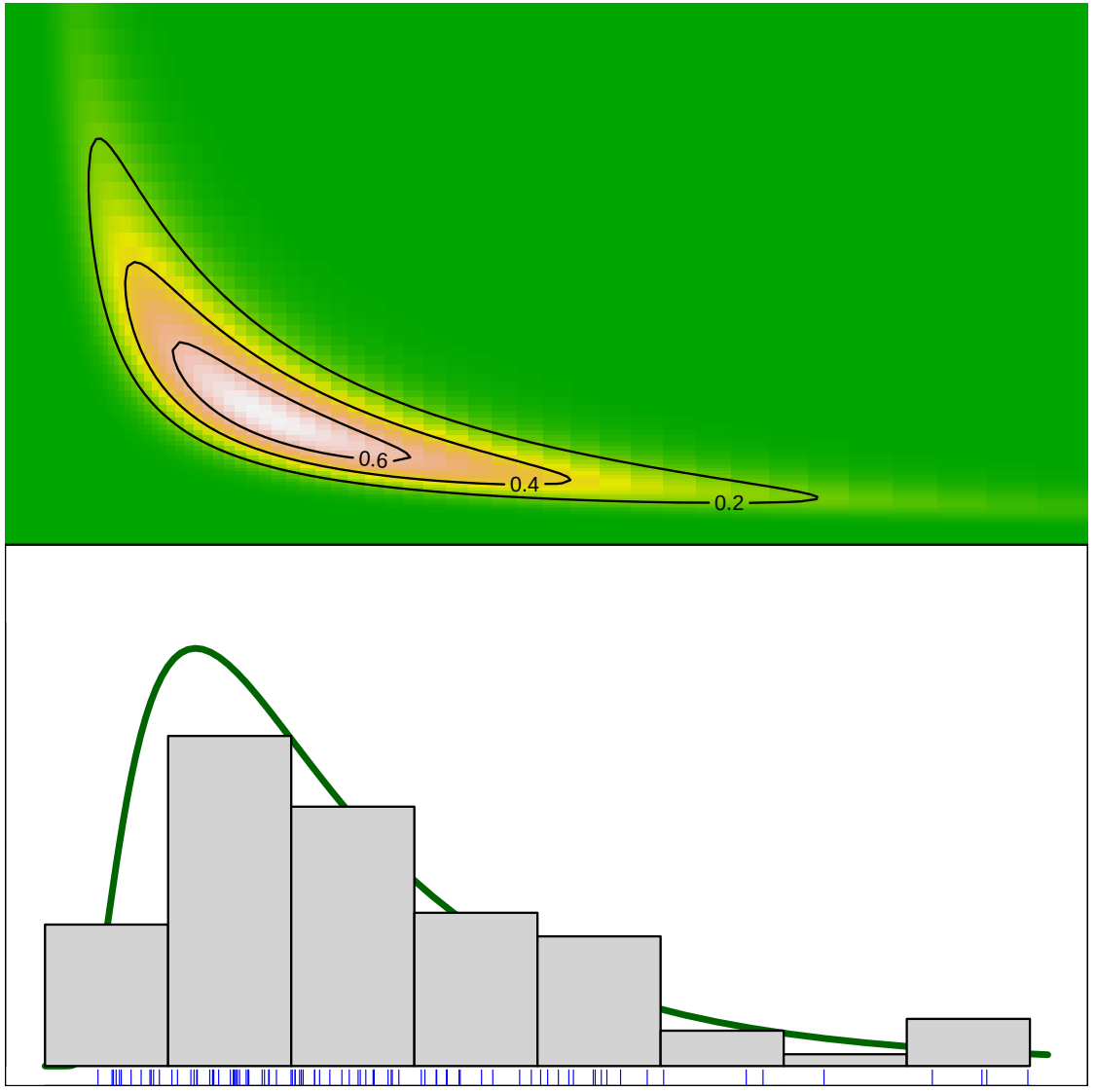
  2: Marginal model is calculated from joint by integration

  3: Marginal is matched to data as always

- Imagine the distribution $D(\theta)$ is described by a likelihood function $L(y, u, \theta)$, then:

  $$L_M(y, \theta) = \int L(y, u, \theta) du$$

  is the marginal likelihood.



Examples where this is the underlying technique

- Random effects are often used:

  Time series models via state-space models Correlation via hidden processes

  Split-plot models Correlated observations within plot

  Repeated measurements Correlated observations within subject

  Over dispersion in Poisson via Negative binomial Patchiness accounted for

  Spatial models Correlation introduced via hidden field

  ...

- Often our only option!
- When something unobserved gives us extra variation or correlated observations.

The Laplace approximation

Reminder: Multivariate normal distribution

The density for a k-dimensional multivariate normal distribution with mean vector μ and covariance matrix Σ is:

$$L(z) = \frac{1}{(2\pi)^{k/2}\sqrt{|\Sigma|}} \exp\left[-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)\right]$$

We write $Z \sim N_k(\mu, \Sigma)$.

The log is:

$$\ell(z) = -\frac{1}{2}\left\{k\log(2\pi|\Sigma|) + (z-\mu)^T\Sigma^{-1}(z-\mu)\right\}$$

General random effects models

The general random effects model can be represented by its likelihood function:

$$L_M(\theta; y) = \int_{\mathbb{R}^q} L(\theta; u, y) du$$

- y is the observed random variables
- u is the q unobserved random variables
- θ is the model parameters to be estimated

The likelihood function L is the joint likelihood of both the observed and the unobserved random variables.

The likelihood function for estimating θ is the marginal likelihood L_M obtained by integrating out the unobserved random variables.

Notice you may already have seen this

- In the Poisson distribution the variance is equal to the mean, which is an assumption that is not always valid.
- Consider the model:

$$Y \sim \text{Pois}(\lambda), \quad \text{where} \quad \lambda \sim \Gamma\left(n, \frac{1-\phi}{\phi}\right) \quad 0 < \phi < 1$$

- It can be shown that:

$$Y \sim \text{Nbinom}(n, \phi)$$

- Notice:
  - No λ in marginal likelihood for Y
  - Analytical integration is not the typical case

General random effects models

- The integral shown on the previous slide is generally difficult to solve if the number of unobserved random variables is more than a few, i.e. for large values of q.
- A large value of q significantly increases the computational demands due to the product rule which states that if an integral is sampled in m points per dimension to evaluate it, the total number of samples needed is m^q, which rapidly becomes infeasible even for a limited number of random effects.
- The likelihood function gives a very broad definition of mixed models: the only requirement for using mixed modeling is to define a joint likelihood function for the model of interest.
- In this way mixed modeling can be applied to any likelihood based statistical modeling.
- Examples of applications are linear mixed models (LMM) and nonlinear mixed models (NLMM), generalized linear mixed models, but also models based on Markov chains, or SDEs.

The Laplace approximation

- We need to calculate the difficult integral

$$L_M(\theta, y) = \int_{\mathbb{R}^q} L(\theta, u, y) du$$

- So we set up an approximation of $\ell(\theta, u, y) = \log L(\theta, u, y)$

$$\ell(\theta, u, y) \approx \ell(\theta, \hat{u}_\theta, y) - \frac{1}{2}(u - \hat{u}_\theta)^t \left(-\ell''_{uu}(\theta, u, y)|_{u=\hat{u}_\theta}\right)(u - \hat{u}_\theta)$$

- Which (for given θ) is the 2. order Taylor approximation around:

$$\hat{u}_\theta = \underset{u}{\text{argmax}}\ L(\theta, u, y)$$

- With this approximation we can calculate:

$$\begin{aligned} L_M(\theta, y) &= \int_{\mathbb{R}^q} L(\theta, u, y) du \\ &\approx \int_{\mathbb{R}^q} e^{\ell(\theta,\hat{u}_\theta,y) - \frac{1}{2}(u-\hat{u}_\theta)^t \left(-\ell''_{uu}(\theta,u,y)|_{u=\hat{u}_\theta}\right)(u-\hat{u}_\theta)} du \\ &= L(\theta, \hat{u}_\theta, y) \int_{\mathbb{R}^q} e^{-\frac{1}{2}(u-\hat{u}_\theta)^t \left(-\ell''_{uu}(\theta,u,y)|_{u=\hat{u}_\theta}\right)(u-\hat{u}_\theta)} du \\ &= L(\theta, \hat{u}_\theta, y) \sqrt{\frac{(2\pi)^q}{|\left(-\ell''_{uu}(\theta, u, y)|_{u=\hat{u}_\theta}\right)|}} \end{aligned}$$

- In the last step we remember the normalizing constant for a multivariate normal, and that $|A^{-1}| = 1/|A|$.
- Taking the logarithm we get:

$$\ell_M(\theta, y) \approx \ell(\theta, \hat{u}_\theta, y) - \frac{1}{2} \log(|\left(-\ell''_{uu}(\theta, u, y)|_{u=\hat{u}_\theta}\right)|) + \frac{q}{2} \log(2\pi)$$

Laplace approximation work flow

0. Initialize θ to some arbitrary value θ_0

1. With current value for θ optimize joint likelihood w.r.t. u to get $\hat{u}_\theta$ and corresponding Hessian $H(\hat{u}_\theta)$.

2. Use $\hat{u}_\theta$ and $H(\hat{u}_\theta)$ to approximate $\ell_M(\theta)$

3. Compute value and gradient of $\ell_M(\theta)$

4. If the gradient is "$>\epsilon$" set θ to a different value and go to **1.**

Notice the huge number of — possibly high dimensional — optimizations that are required.

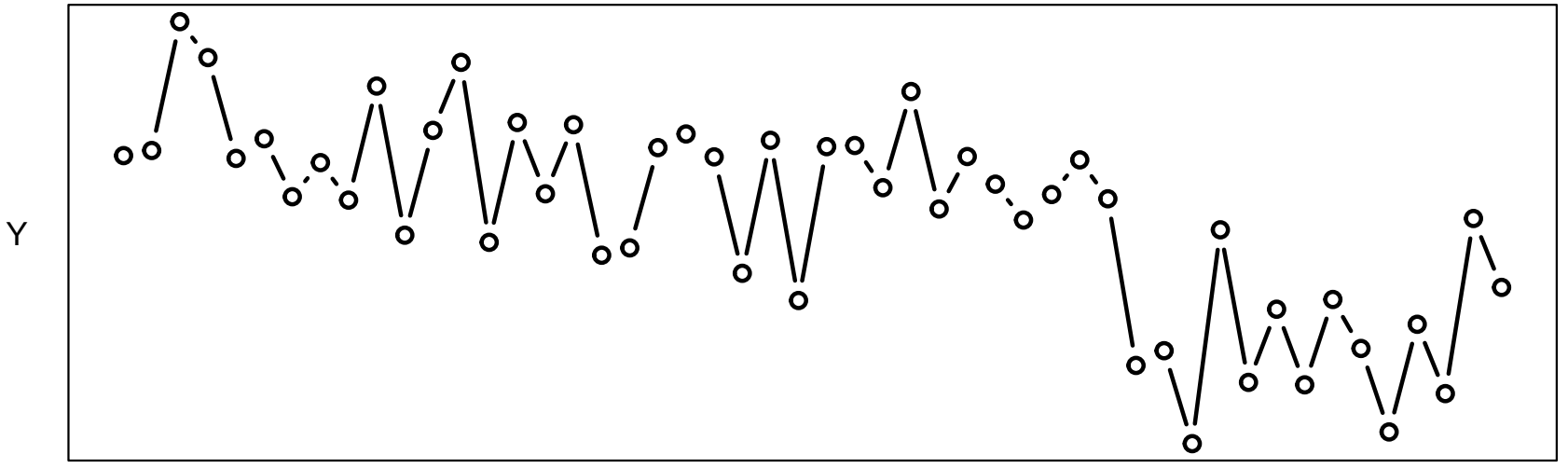
The Laplace approximation

- The Laplace likelihood only approximates the marginal likelihood for mixed models with nonlinear random effects and thus maximizing the Laplace likelihood will result in some amount of error in the resulting estimates.
- It can be shown that joint log-likelihood converges to a quadratic function of the random effect for increasing number of observations per random effect and thus that the Laplace approximation is asymptotically exact.
- In practical applications the accuracy of the Laplace approximation may still be of concern, but often improved numerical approximation of the marginal likelihood (such as Gaussian quadrature) may easily be computationally infeasible to perform.
- We can exaluate if it is causing bias (subject for later)
- Another option for improving the accuracy is importance sampling.

Simple state-space model

State-space model (RW+noise)

- Observation vector Y generated from:
  - $\lambda_i = \lambda_{i-1} + \eta_i$
  - $Y_i = \lambda_i + \varepsilon_i$
  - where $i = 1 \ldots 50$, $\eta_i \sim N(0, \sigma_\lambda^2)$, and $\varepsilon_i \sim N(0, \sigma_Y^2)$ all independent.



- Notice λ vector unobserved.
- Here we wish to estimate both λ and the model parameters ($\lambda_\circ, \sigma_\lambda$, and σ_ε)

Basic Kalman Filter

```
library(RTMB)
dat <- list(y=scan("kf.dat"))
par <- list(logSdRw=0, logSdObs=0, lam0=0)

f<-function(par){
  getAll(par,dat)

  N <- length(y)
  lam <- numeric(N)
  lamVar <- numeric(N)
  lamPred <- numeric(N)
  lamPredVar <- numeric(N)
  lamSmooth <- numeric(N)
  lamSmoothVar <- numeric(N)
  yPredVar <- numeric(N)
  w <- numeric(N)
  nll <- 0
  varLam <- exp(2.0*logSdRw)
  varY <- exp(2.0*logSdObs)
  lamPred[1] <- lam0
  lamPredVar[1] <- varLam
  yPredVar[1] <- lamPredVar[1]+varY
  w[1] <- y[1]-lamPred[1]
  lam[1] <- lamPred[1]+lamPredVar[1]/yPredVar[1]*w[1]
  lamVar[1] <- lamPredVar[1]-lamPredVar[1]/yPredVar[1]*lamPredVar[1]
  nll = nll-dnorm(w[1], sd=sqrt(yPredVar[1]),log=TRUE)
```

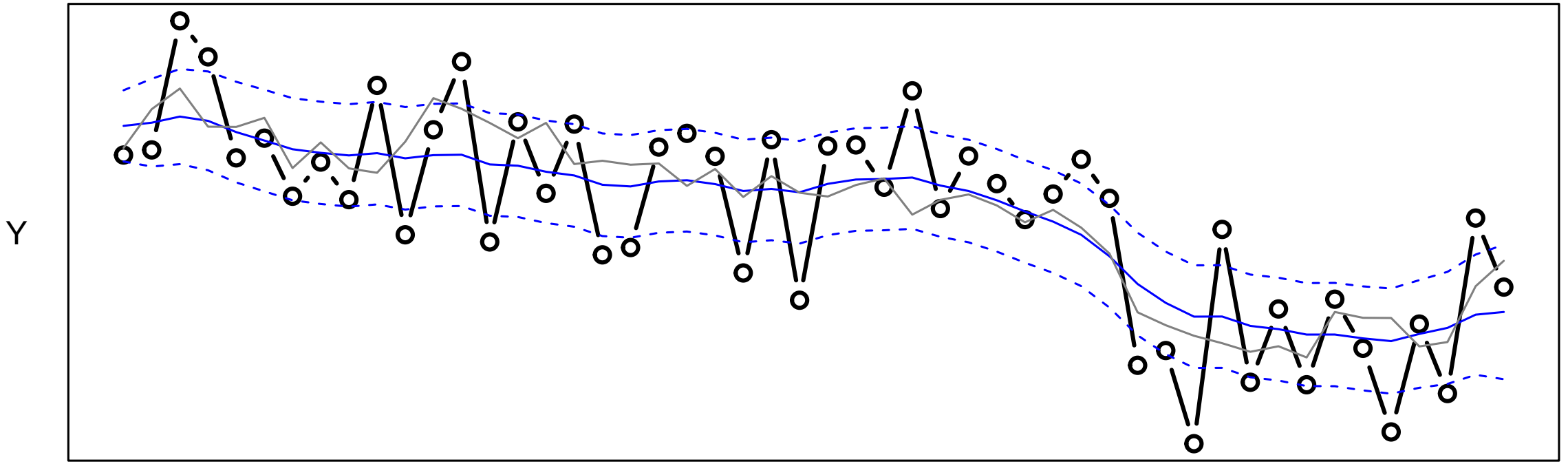
kf.R

... and there is more

```
  for(i in 2:N){
    lamPred[i] <- lam[i-1]
    lamPredVar[i] <- lamVar[i-1]+varLam
    yPredVar[i] <- lamPredVar[i]+varY
    w[i] <- y[i]-lamPred[i]
    lam[i] <- lamPred[i]+lamPredVar[i]/yPredVar[i]*w[i]
    lamVar[i] <- lamPredVar[i]-lamPredVar[i]/yPredVar[i]*lamPredVar[i]
    nll <- nll-dnorm(w[i],sd=sqrt(yPredVar[i]), log=TRUE)
  }
  # ----------------- smoothing part
  lamSmooth[N] <- lam[N]
  lamSmoothVar[N] <- lamVar[N]
  for(i in (N-1):1){
    varFrac <- lamVar[i]/lamPredVar[i+1]
    lamSmooth[i] <- lam[i]+varFrac*(lamSmooth[i+1]-lamPred[i+1])
    lamSmoothVar[i] <- lamVar[i]+varFrac*(lamSmoothVar[i+1]-lamPredVar[i+1])*varFrac
  }
  REPORT(lamSmooth)
  REPORT(lamSmoothVar)
  nll
}

obj<-MakeADFun(f,par)

obj$fn()
obj$gr()
opt<-nlminb(obj$par,obj$fn,obj$gr)

rep<-obj$report()

save(rep,file="kf.RData")
```

kf.R

Estimates from the Kalman Filter



- Fast for linear and Gaussian state-space models
- Must be extended to deal with non-linear models
- Non-Gaussian models are difficult and need either simulation based filters or discrete approximation of state-space (only possible in low dimension)

But we can construct the joint likelihood

- The joint likelihood contributions from the random effects:

$$(\lambda_1 - \lambda_\circ) \sim N(0, \sigma_\lambda^2)$$
$$(\lambda_2 - \lambda_1) \sim N(0, \sigma_\lambda^2)$$
$$\ldots$$
$$(\lambda_{50} - \lambda_{49}) \sim N(0, \sigma_\lambda^2)$$

- The joint likelihood contributions from the observations:

$$y_1 \sim N(\lambda_1, \sigma^2)$$
$$y_2 \sim N(\lambda_2, \sigma^2)$$
$$\ldots$$
$$y_{50} \sim N(\lambda_{50}, \sigma^2)$$

- So the joint negative log likelihood becomes:

$$-\sum_{i=1}^{50} \log(\varphi_{\circ,\sigma_\lambda}(\lambda_i - \lambda_{i-1})) - \sum_{i=1}^{50} \log(\varphi_{\lambda_i,\sigma}(y_i))$$

We can solve this by Laplace approximation in RTMB

- Code up the joint negative log likelihood
  - Declare the unobserved quantities as 'random' (but part of parameter set)
  - Code as if the unobserved were observed
- Identify which quantities are to be considered as random effects (unobserved), as e.g:

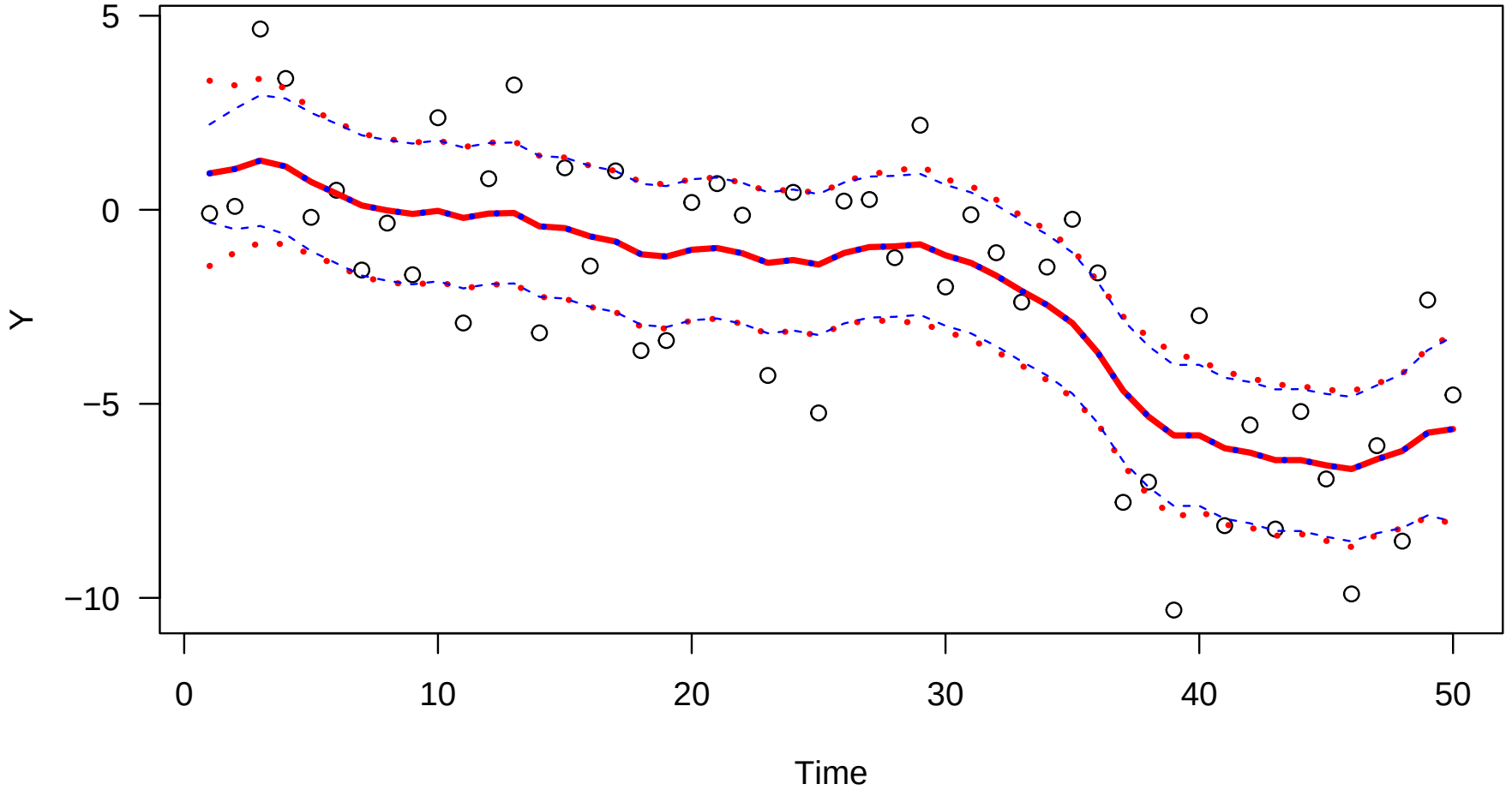
```
obj <- MakeADFun(nll,par,random="lam")
```

Random walk + noise in RTMB

```
library(RTMB)
dat <- list(y=scan("Y.dat"))
par <- list(logSdRw=0, logSdObs=0, lam0=0, lam=numeric(length(dat$y)))

jnll<-function(par){
  getAll(par, dat)
  sdRw <- exp(logSdRw)
  sdObs <- exp(logSdObs)
  ret <- -dnorm(lam[1], mean=lam0, sd=sdRw, log=TRUE)
  ret <- ret - sum(dnorm(diff(lam), sd=sdRw, log=TRUE))
  ret <- ret - sum(dnorm(y, mean=lam, sd=sdObs, log=TRUE))
  ret
}

obj <- MakeADFun(jnll, par, random="lam", silent=TRUE)
opt <- nlminb(obj$par, obj$fn, obj$gr)

sdr <- sdreport(obj)
pl <- as.list(sdr, "Est")
plsd <- as.list(sdr, "Std")

plot(dat$y, xlab="Time", ylab="Y", las=1)
lines(pl$lam, lwd=3, col="red")
lines(pl$lam-2*plsd$lam, lty="dotted", lwd=3, col="red")
lines(pl$lam+2*plsd$lam, lty="dotted", lwd=3, col="red")
```

ss.R

Did we get the same answer?



Same estimated track, but uncertainties differ a little. Why? Fixable?

Exercise 1: RW + noise with missing observations

a) Consider again the simple random walk plus noise example. In the file: `rwmissing.dat` six of the observations are missing (coded as 'NA'). Consider and implement changes in the program to deal with that.

b) Make a plot of the estimated process and its confidence interval (similar to the one on the previous slide) to illustrate the effect of the missing observations.

Hint: The following code can be used to identify which observations are not missing:

```
idx <- which(!is.na(y))
```

That's basically it(!)

- Seems like a too simple example
- This approach can be useful very generally, e.g:
- Higher dimension
- Some non-linearity
- Some non-normality in the process and observations
- So let's look at a few examples...

Higher dimension

- Consider the model:
  - The unobserved part:

$$\lambda_i^{(1)} = \lambda_{i-1}^{(1)} + \varepsilon_i^{(1)}$$
$$\lambda_i^{(2)} = \lambda_{i-1}^{(2)} + \varepsilon_i^{(2)}$$
$$\lambda_i^{(3)} = \lambda_{i-1}^{(3)} + \varepsilon_i^{(3)}$$

  - The observed part:

$$y_i^{(1)} = \lambda_i^{(1)} + \eta_i^{(1)}$$
$$y_i^{(2)} = \lambda_i^{(2)} + \eta_i^{(2)}$$
$$y_i^{(3)} = \lambda_i^{(3)} + \eta_i^{(3)}$$

  - Let's start by all noise terms have separate variance and are assumed Gaussian and independent.

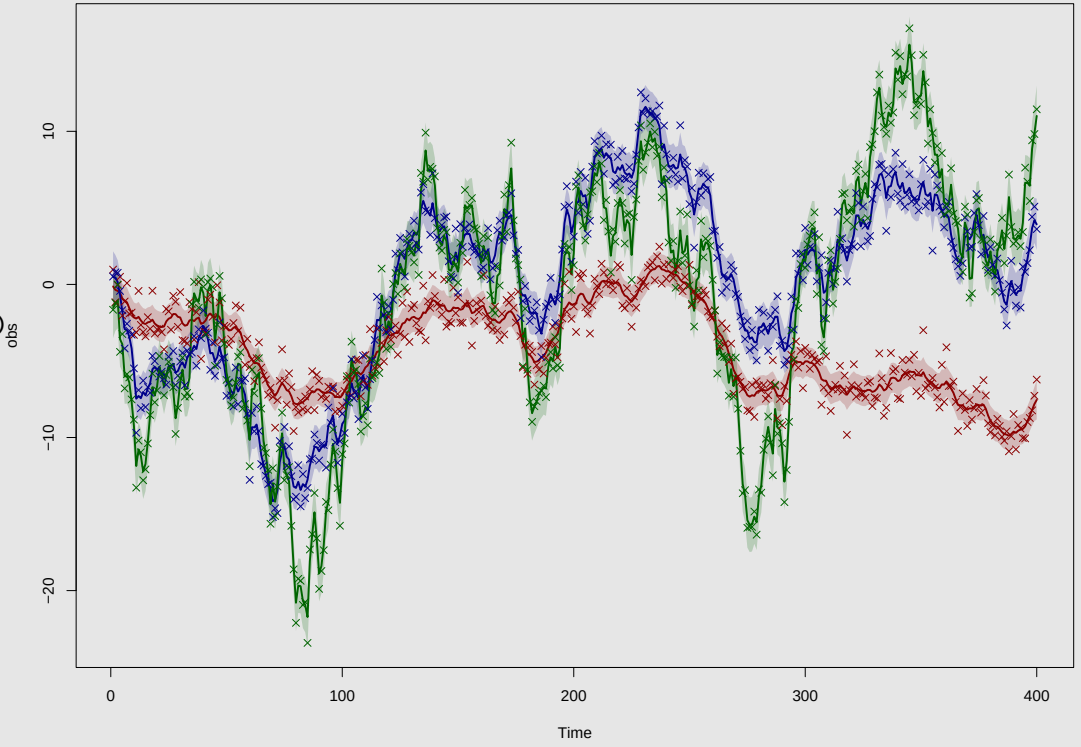
The code could be:

```
library(RTMB)
obs <- read.table("files/mvrw.dat", header=FALSE)
par <- list(logSdLam=rep(0,ncol(obs)), logSdObs=rep(0,ncol(obs)), lambda=matrix(0, nrow=nrow(obs), ncol=ncol(obs)))

f<-function(par){
  getAll(par)
  stateDim <- ncol(lambda)
  timeSteps <- nrow(lambda)
  sdLam <- exp(logSdLam)
  sdObs <- exp(logSdObs);
  S <- diag(sdLam^2)
  Q <- diag(sdObs^2)
  ret <- 0
  for(i in 2:timeSteps){
    ret <- ret - dmvnorm(lambda[i,], lambda[i-1,], Sigma=S, log=TRUE)
  }
  for(i in 1:timeSteps){
    ret <- ret - dmvnorm(obs[i,], lambda[i,], Sigma=Q, log=TRUE)
  }
  ret
}

obj <- MakeADFun(f,par,random="lambda")
opt<-nlminb(obj$par,obj$fn,obj$gr)
```

files/mvrw.R

... or even:

```
library(RTMB)
obs <- read.table("files/mvrw.dat", header=FALSE)
par <- list(logSdLam=rep(0,ncol(obs)), logSdObs=rep(0,ncol(obs)), lambda=matrix(0, nrow=nrow(obs), ncol=ncol(obs)))

f <- function(par){
  getAll(par)
  stateDim <- ncol(lambda)
  timeSteps <- nrow(lambda)
  sdLam <- exp(logSdLam)
  sdObs <- exp(logSdObs);
  S <- diag(sdLam^2)
  Q <- diag(sdObs^2)
  ret <- -sum(dmvnorm(lambda[-1,], lambda[-nrow(lambda),], Sigma=S, log=TRUE))
  ret <- ret - sum(dmvnorm(obs, lambda, Sigma=Q, log=TRUE))
  ret
}
obj <- MakeADFun(f,par,random="lambda", silent=TRUE)
opt <- nlminb(obj$par,obj$fn,obj$gr)
sdr <- sdreport(obj)
pl <- as.list(sdr,"Est")
plsd <- as.list(sdr,"Std")

Time <- 1:nrow(obs)
matplot(Time, obs, type="n")
for(j in 1:ncol(obs))polygon(cbind(c(Time,rev(Time)), c(pl$lambda[,j]-2*plsd$lambda[,j], rev(pl$lambda[,j]+2*plsd$lambda[,j]))),
                             col=scales:::alpha(palette()[j],.2), border=NA)
matplot(pl$lambda, type="l", lwd=2, add=TRUE, lty="solid")
matplot(Time, obs, pch=4, add=TRUE)
```



files/mvrw2.R

Exercise: Correlation between the processes

- In the multivariate random walk example the three different λ processes were assumed independent. If $\lambda^{(1)}$ and $\lambda^{(2)}$ are two environmental indices and $\lambda^{(3)}$ is abundance of our most economically important fish stock, then we may be interested in estimating how correlated they are. Consider the model:
- Define $\Delta\lambda_t = \begin{pmatrix} \lambda_t^{(1)} - \lambda_{t-1}^{(1)} \\ \lambda_t^{(2)} - \lambda_{t-1}^{(2)} \\ \lambda_t^{(3)} - \lambda_{t-1}^{(3)} \end{pmatrix}$
- Let's assume $\Delta\lambda_t \sim N_{3\times 3}(0, \Sigma)$ where: $\Sigma_{i,j} = \rho^{|i-j|}\sigma_i\sigma_j$
- Here ρ is a new model parameter, which should be bounded between -1 and 1.

Hint: The following function is helpful for bounding within (-1,1).

```
bound <- function(x)2*plogis(x)-1
```

Non-linear and multivariate state-space model

- The unobserved part:

$$\lambda_i^{(1)} = \lambda_{i-1}^{(1)} + \varepsilon_i^{(1)}$$
$$\lambda_i^{(2)} = \phi\lambda_{i-1}^{(2)} + \varepsilon_i^{(2)}$$

- The observed part:

$$y_i^{(1)} = e^{\lambda_i^{(1)}}/(1 + e^{\lambda_i^{(1)}}) + \eta_i^{(1)}$$
$$y_i^{(2)} = \lambda_i^{(2)} + \eta_i^{(2)}$$
$$y_i^{(3)} = \lambda_i^{(1)}\lambda_i^{(2)} + \eta_i^{(3)}$$

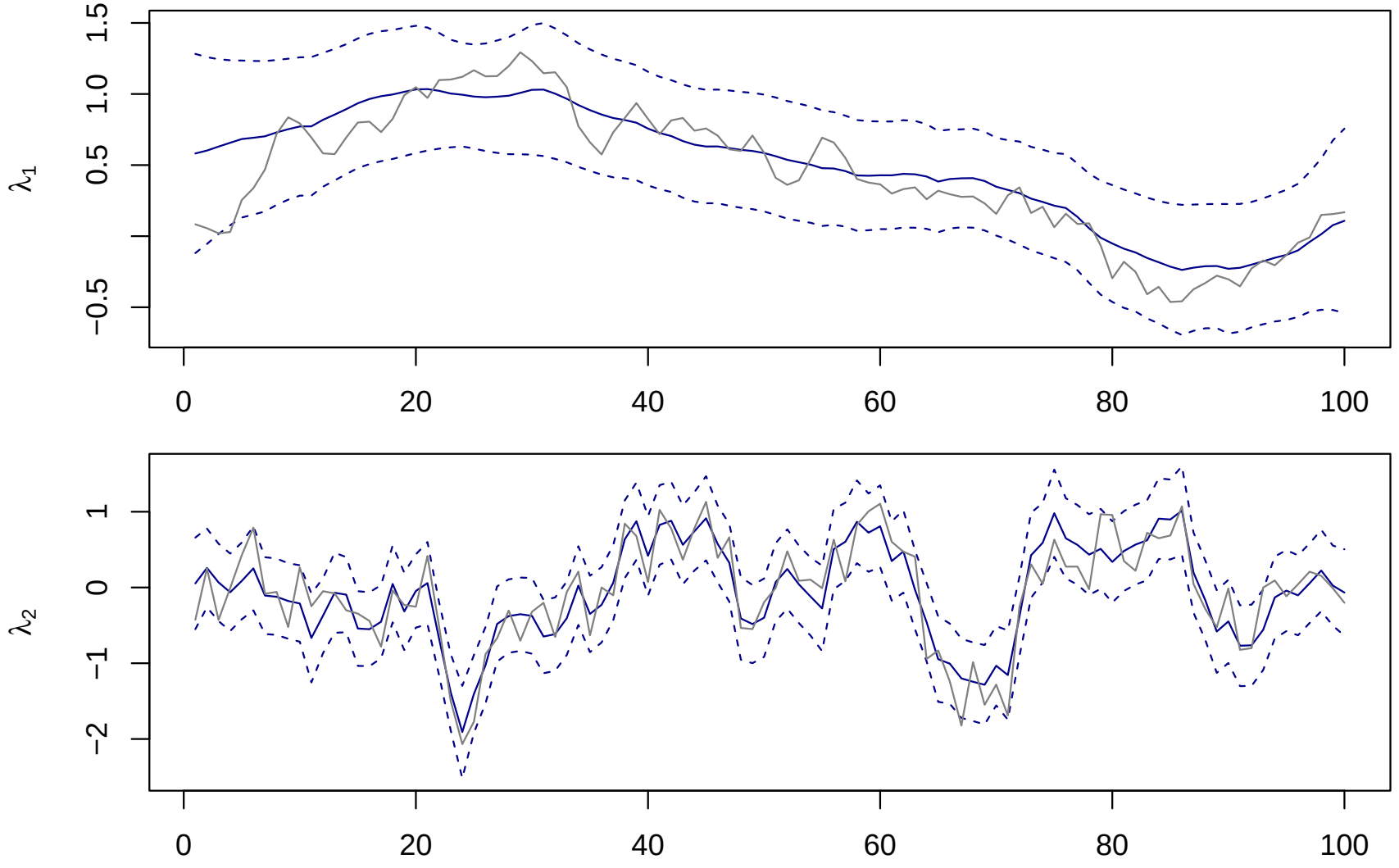
- All noise terms have separate variance and are independent.

Non-linear and multivariate state-space model

```
library(RTMB)
obs <- as.matrix(read.table("nonlin.dat", header=FALSE))
par <- list(trans_phi=0, logsdLam=rep(0,2), logsdObs=rep(0,ncol(obs)), lam=matrix(0,nrow=nrow(obs),ncol=2))

f<-function(par){
  getAll(par)
  timeSteps <- nrow(lam)
  sdLam <- exp(logsdLam)
  sdObs <- exp(logsdObs)
  phi <- 2*plogis(trans_phi)-1

  jnll <- 0
  for(i in 2:timeSteps){
    predU <- c(lam[i-1,1], phi*lam[i-1,2])
    jnll <- jnll - sum(dnorm(lam[i,],predU,sdLam,log=TRUE))
  }
  for(i in 1:timeSteps){
    predObs <- c(exp(lam[i,1])/(1+exp(lam[i,1])), lam[i,2], lam[i,1]*lam[i,2])
    jnll <- jnll -sum(dnorm(obs[i,],predObs,sdObs,log=TRUE))
  }
  jnll
}
obj <- MakeADFun(f,par,random="lam", silent=TRUE)
opt <- nlminb(obj$par,obj$fn,obj$gr)

sdr <- sdreport(obj)
pl <- as.list(sdr,"Est")
plsd <- as.list(sdr,"Std")
```

files/nl.R

Non-linear and multivariate state-space model result



Discrete valued time series

- Examples from Kuk & Cheng (1999).
- The model:

$$y_i \sim \text{Pois}(\lambda_i) \ , \ \text{where}$$
$$\log(\lambda_i) = X_i b + u_i \ , \ \text{and}$$
$$u_i = a u_{i-1} + \varepsilon_i$$

Here, X_i is a design matrix of covariates, b is a vector of regression parameters and u_i is an AR(1). The dimension of b is 6 and i=1,. . . ,168.

	β_1	β_2	β_3	β_4	β_5	β_6	a	σ
RTMB	0.242	-3.814	0.162	-0.482	0.413	-0.011	0.627	0.538
ADMB-RE	0.242	-3.814	0.162	-0.482	0.413	-0.011	0.627	0.538
Std. dev.	0.270	2.76	0.15	0.16	0.13	0.13	0.19	0.15
Kuk & Cheng	0.244	-3.82	0.162	-0.478	0.413	-0.0109	0.665	0.519

Discrete valued time series code

```
library(RTMB)
df <- read.table("pol.dat", header=FALSE)
dat <- list(y=df[,1], X=as.matrix(df[,-1]))
par <- list(b=numeric(ncol(dat$X)), trans_a=0, logSigma=0, u=numeric(length(dat$y)))

f <- function(par){
  getAll(dat,par)
  timeSteps <- length(y)
  a <- 2*plogis(trans_a)-1
  sigma <- exp(logSigma)
  jnll <- -dnorm(u[1], 0, sigma/sqrt(1-a*a),log=TRUE)
  for(i in 2:timeSteps){
    jnll <- jnll - dnorm(u[i],a*u[i-1],sigma,log=TRUE)
  }
  eta <- X%*%b
  for(i in 1:timeSteps){
    lambda <- exp(eta[i]+u[i])
    jnll <- jnll - dpois(y[i],lambda,log=TRUE)
  }
  jnll
}

obj <- MakeADFun(f,par,random="u")
opt<-nlminb(obj$par,obj$fn,obj$gr)

sdr<-sdreport(obj)
pl <- as.list(sdr,"Est")
plsd <- as.list(sdr,"Std")
```

files/pol.R

Exercise 3: A theta logistic population model

A theta logistic population model is defined for the log-transformed population size as a nonlinear function of its previous size in the following way:

$$X_t = X_{t-1} + r_0 \left(1 - \left(\frac{\exp(X_{t-1})}{K}\right)^{\theta}\right) + e_t,$$

$$Y_t = X_t + u_t,$$

where $e_t \sim N(0, Q)$ and $u_t \sim N(0, R)$.

Data for this model is available in the file: `theta.dat`

a) Fit the model to data and estimate the five model parameters.

Hint: It is helpful to log transform the parameters and initialize $\log(K)$ to around 6.