Error model estimation by maximum-likelihood methods

in the context of dynamic modeling

Mirjam Fehling-Kaschek[1], Daniel Kaschek[1], Wolfgang Mader[1], Marcus Rosenblatt[1], Jens Timmer[1,2,3]
[1]Institute of Physics, Freiburg University
[2]BIOSS Centre for Biological Signalling Studies, Freiburg
[3]Freiburg Center for Systems Biology (ZBSA), Freiburg University

UNI FREIBURG

Introduction

Common Setup:

- time series data with $n \leq 4$ replicates
- use mean values for modelling
- empirical variances bad estimate for uncertainties

Goal:

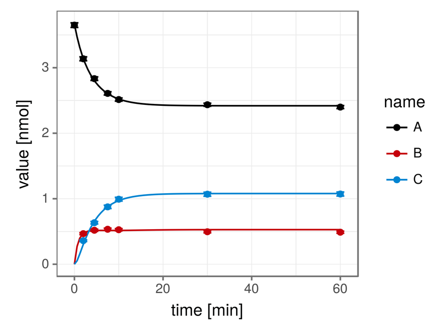
- reliable estimate of measurement uncertainties

Implementations:

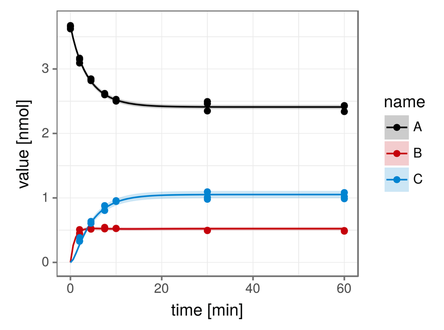
1: standalone fit of error model (data preprocessing) $\rightarrow$ mean-variance tuples

2: simultaneous fit of model and error parameters $\rightarrow$ model residuals

1. preprocessed data:



2. combined fit:



dMod: dynamic modeling in R

Model: dynamic system $\dot{x} = f(x, p, u)$ with states x, parameters p, forcings $u(t)$.

Model Setup: example

$$A \rightarrow B \rightarrow \emptyset$$

```
Specify the model reactions:
 > f <- addReaction(NULL, "A", "B", "k1*A", "translation")
 > f <- addReaction(f, "B", "", "-k2*B","degradation")
De ne observables:
 > obs <- eqnvec(Bobs = "s1*B")
Parameter transformations:
 > innerpars <- getSymbols(c(names(f), f, obs))
 > trafo <- repar("x ~ exp(x)", x = innerpars)
 > # condition specific:
 > conditions <- c("a", "b")
 > trafoL <- list(
     a = replaceSymbols("s", "sa", trafo),
     b = replaceSymbols("s", "sb", trafo))
Generate compiled C code for ODEs and Sensitivities:
 > model <- odemodel(f)
 > # prediction function
 > x <- Xs(model)
 > # Observation function
 > g <- Y(obs, x, compile = TRUE, modelname = "obsfn")
 > # Condition-specific transformation function
 > p <- NULL
 > for (C in conditions) {
       p <- p + P(trafo[[C]], condition = C)
   }
 > #Calculate a prediction:
 > pars <- c(A=5, B=0, k1=1, k2=0.2, sa=1, sb=2)
 > pred <- (g*x*p)(times = 0:50, pars = pars)
```

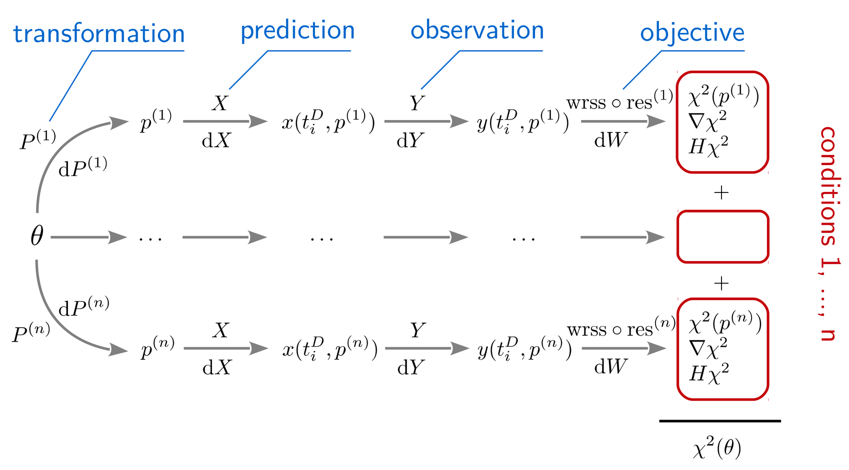
Data: time-series

name	time	value	sigma
Bobs	0	0.3	0.1
Bobs	20	1.2	0.2
Bobs	50	4.1	0.3

Parameter Estimation:

$$-2 \log L(\theta) = \sum_i \left(\frac{x_i(\theta) - x_i^D}{\sigma_i(\theta)} \right)^2$$

```
Objective function:
 > obj <- normL2(data, g*x*p)
Optimization via trust package:
 > myfit <- trust(obj, c(A=5,B=0,k1=1,k2=1,sa=1,sb=1))
```



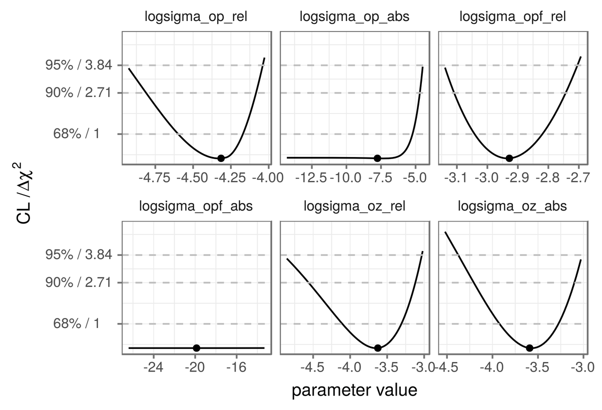
2. Combined Fitting

Procedure: simultaneous estimation of dynamic model and error parameters

$$-2 \log L(\theta) = \sum_i \left(\frac{x_i(\theta) - x_i^D}{\sigma_i(\theta)} \right)^2 + \log(\sigma_i(\theta)^2)$$

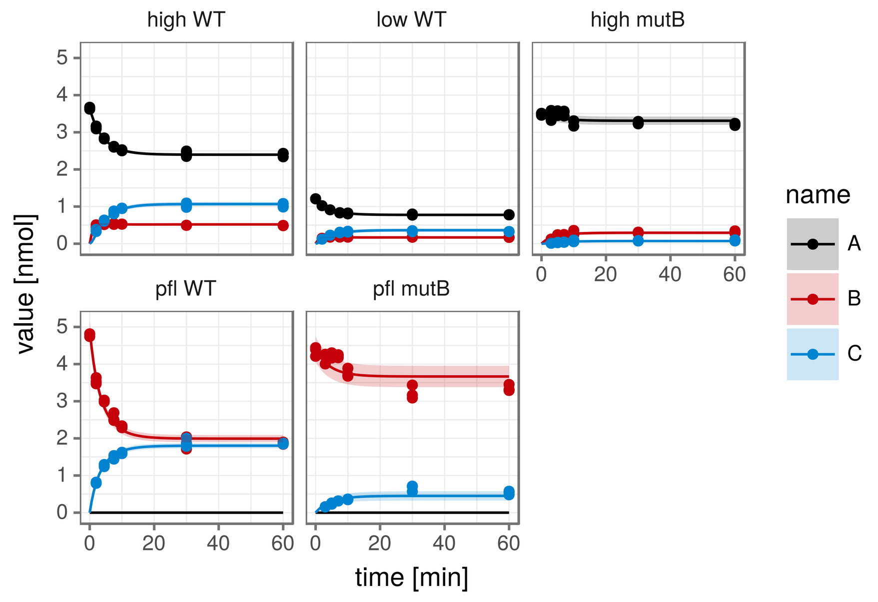
Error Model:

```
 > err <- eqnvec(Bobs = "sqrt(sig_rel^2*Bobs^2+sig_abs^2)")
 > e <- Y(err, g, compile = TRUE, modelname = "errfn")
Adjust parameters, transformations, objective function, e.g.:
 > innerpars <- getParameters(model, g, e)
 > obj <- normL2(data, g*x*p, e)
```



- in general: σ_{rel}, σ_{abs} per observable
- model identification: profile likelihood

Result: Combined fit with bands from error model



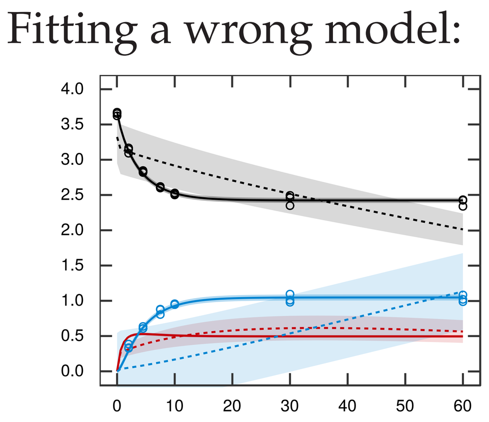
Pros:

- applicable for $n = 1$
- capture all relevant uncertainties

Cons:

- no information from absolute χ^2 value, but comparison of nested models possible
- error model parameters increased to compensate for bad models
- likelihood fit underestimates errors

Fitting a wrong model:



Outlook:

- unbiased variance estimation by correction for degrees of freedom

1. Standalone Fit of Error Model

Estimate error from measured variances:

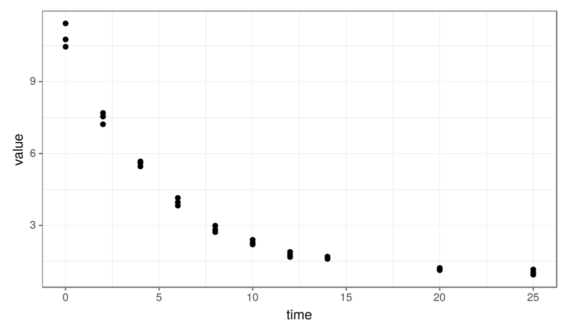
- $\rho = \frac{(n-1)v}{\sigma^2} \propto \chi^2_{n-1}$
- PDF: $\Phi_{n-1}(\rho) \rightarrow \Phi_{n-1}(v)$
  $\Phi_{n-1}(\rho(v)) \frac{d\rho}{dv} dv = \Phi_{n-1}\left(\frac{(n-1)v}{\sigma^2}\right) \frac{(n-1)}{\sigma^2} dv$
- estimate σ via log-likelihood method:

$$2l(\sigma_s) = \sum_i \log\left[\hat{\sigma}^2(x_i, \sigma_s)\right] - \log\left[\Phi_{n_i-1}\left(\frac{(n_i-1)v_i}{\hat{\sigma}^2(x_i, \sigma_s)}\right)\right]$$

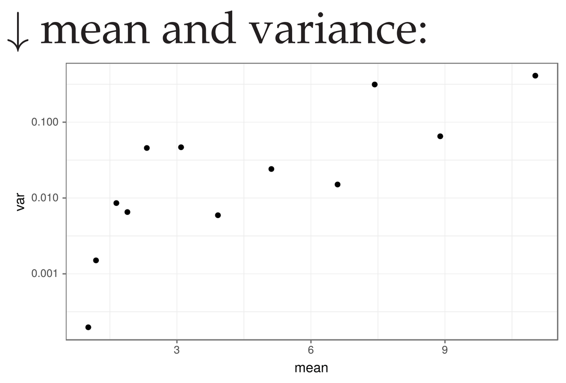
- $\hat{\sigma}^2(x_D, \sigma_s)$: error model with parameters σ_s

$$\hat{\sigma}^2(x_D) = \sigma_0^2 + \sigma_{rel}^2 x_D^2$$

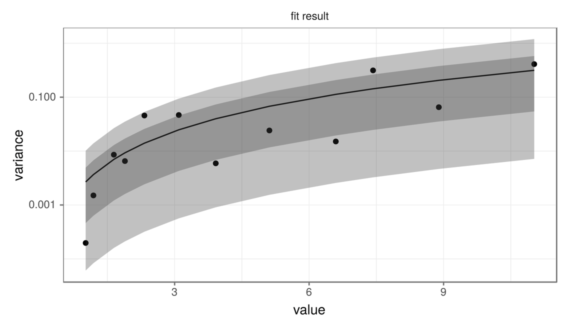
timeseries:



$\downarrow$ mean and variance:

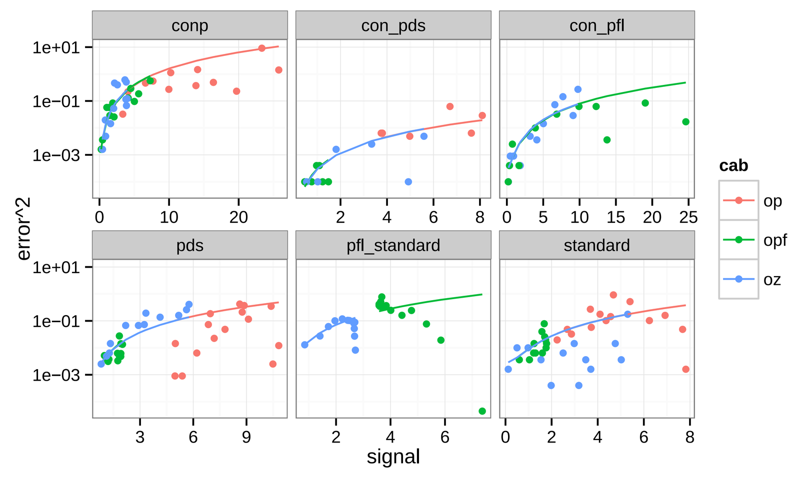


$\downarrow$ error model:



Implementation:

```
Get mean values and variances:
 > data <- reduceReplicates("data.csv")
Fit error model:
 > errorModel <- "exp(s0)+exp(srel)*x^2"
 > factors <- "conditions"
 > pars <- c(s0=1,srel=0.1)
 > data <- fitErrorModel(data, factors, errorModel, pars)
```

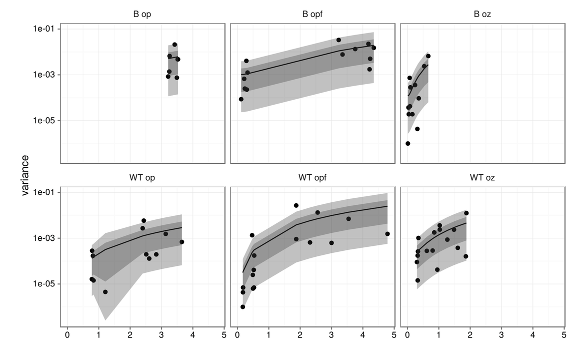


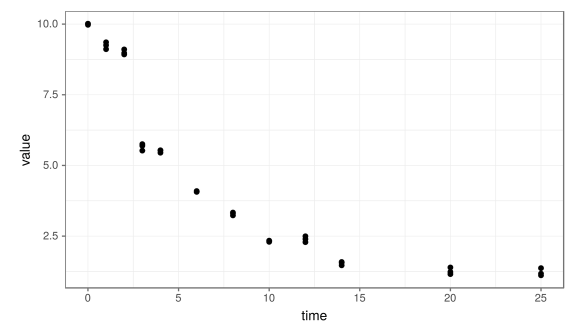
Error Model Parametrisation:

- specify common and independent parameters for data subsets: $\rightarrow$ per dataset, observable, ...
- $n = 1$: use parameters of similar data
- model identification: comparison of fits

Pro & Cons:

- independent of model fit $\rightarrow$ no overcompensation due to wrong model
- easy validation of error model: $\rightarrow$ plot of fit result with sigma bands
- error model might underestimate error: systematics not captured by replicates $\rightarrow$ fit of dynamic model: model parameter uncertainties underestimated

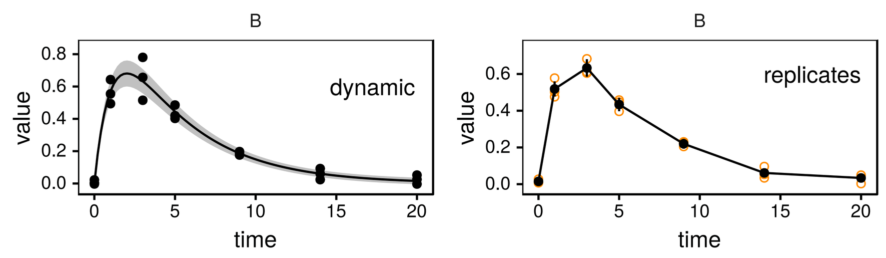


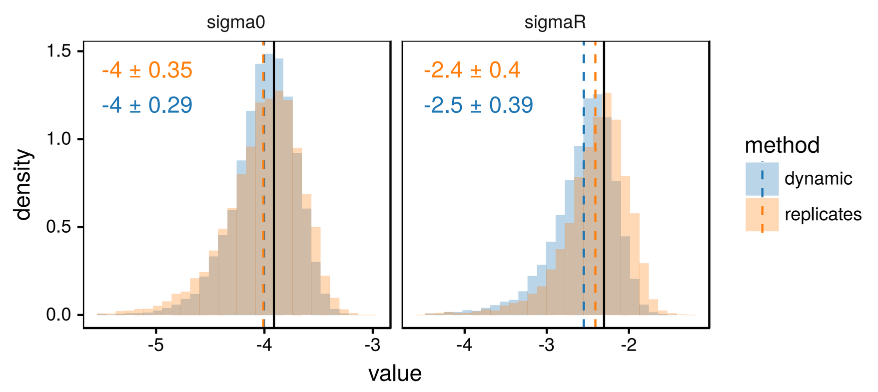


Comparison

Setup:

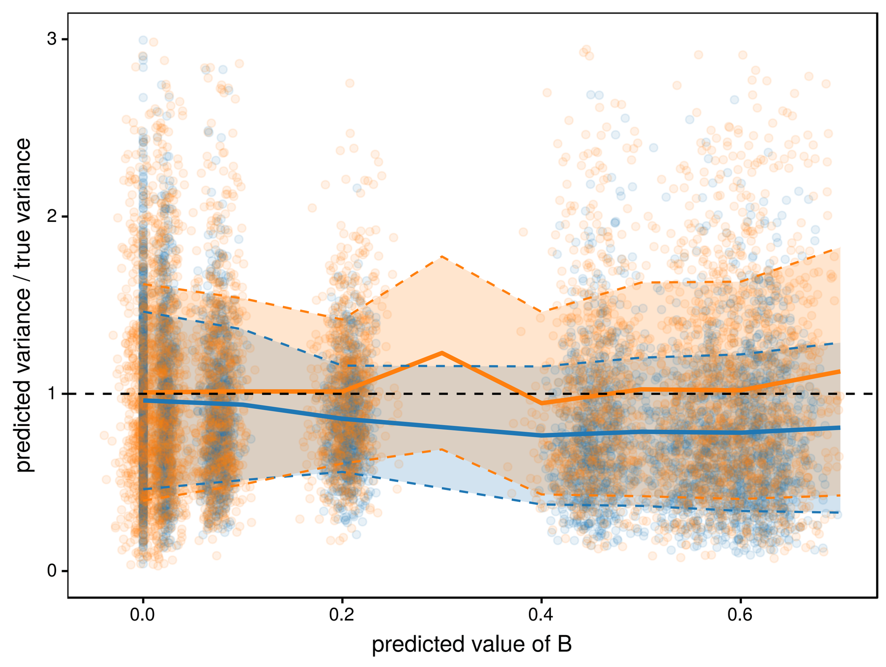
- $A \rightarrow B \rightarrow C$ model observables: B
- simulate with: $\sigma_0 = 0.02, \sigma_R = 0.1$
- fit on log-scale: $\sigma_0^{\log} = -3.9, \sigma_R^{\log} = -2.3$





Results:

- similar performance by both methods: comparable width of σ_0, σ_R distributions
- 1. replicates: variance prediction unbiased
  2. dynamic: underestimation of the variance



Conclusions:

- typical example of bias-variace tradeoff

Bundesministerium für Bildung und Forschung

Brigitte-Schlieben-Lange-Programm

bioss CENTRE FOR BIOLOGICAL SIGNALLING STUDIES

zbsa zentrum für biosystemanalyse