Who Ruins Our Fixed Effect?
and How To Fix It?

Junshui Ma and **Vladimir Svetnik**

Biometrics Research, Merck & Co. Inc.

NISS 2014

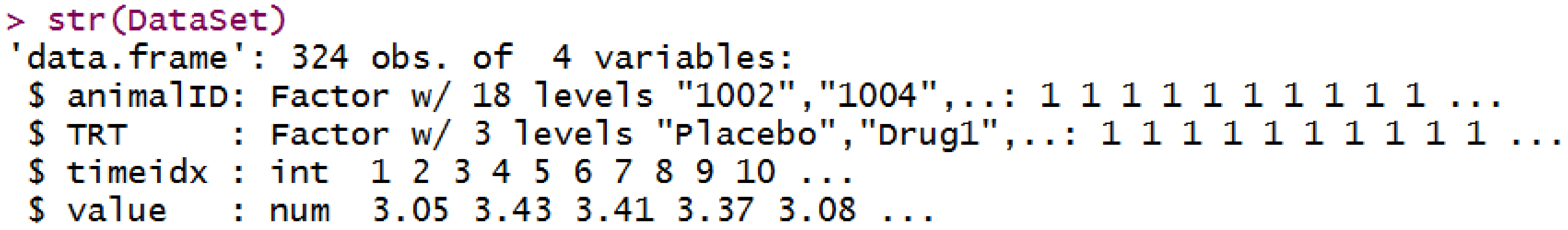
Acknowledgements

- Dr. Richard Raubertas and other colleagues of mine.
- Scientists who provided the data.

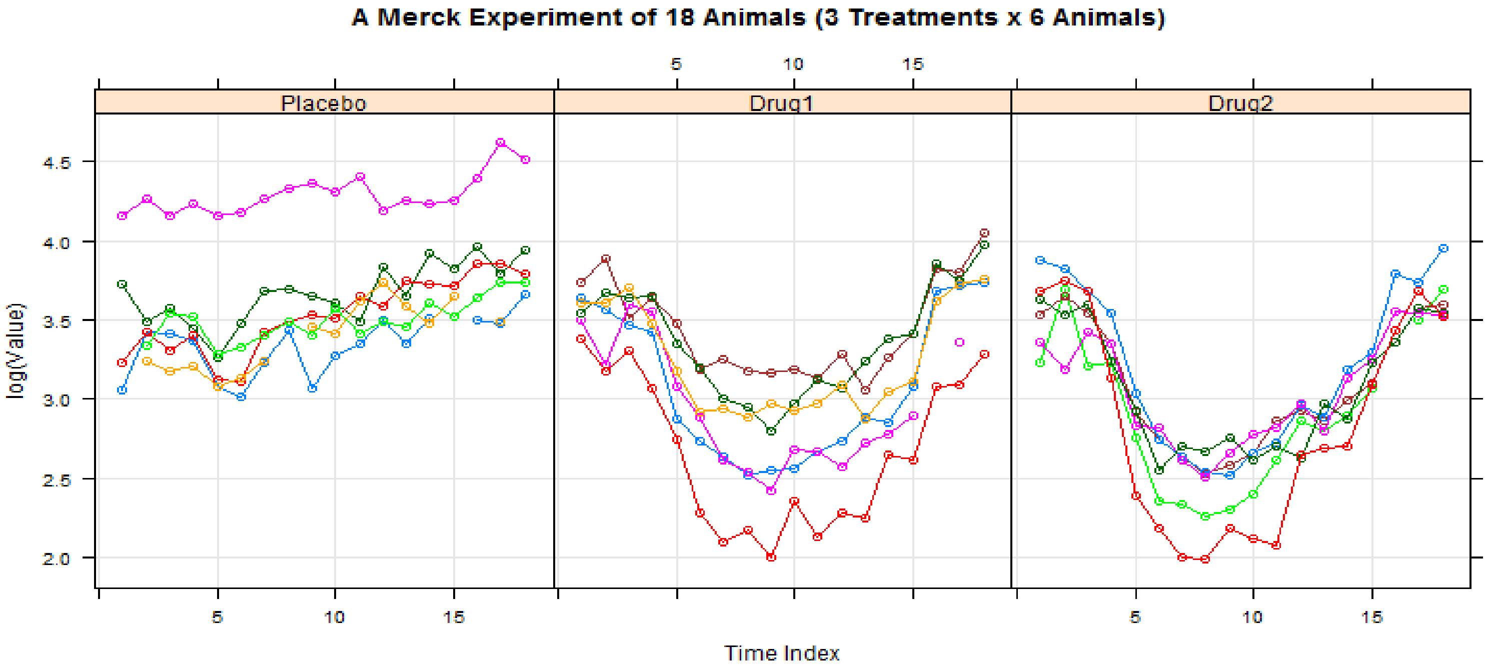
Outlines

- **The bias problem**
- A Bayesian solution
- A Frequentist’s solution
- Summary

The Example Dataset (i.e. Rich’s Dataset)



```
> str(DataSet)
'data.frame':	324 obs. of  4 variables:
 $ animalID: Factor w/ 18 levels "1002","1004",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ TRT     : Factor w/ 3 levels "Placebo","Drug1",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ timeidx : int  1 2 3 4 5 6 7 8 9 10 ...
 $ value   : num  3.05 3.43 3.41 3.37 3.08 ...
```



- 3 treatment (**TRT**) groups, i.e. *Placebo, Drug1,* and *Drug2*.
- 6 animals (**animalID**) in each treatment group, and totally 18 animals.
- 18 values of each animal is measured in 18 time points (**timeidx**).
- A functional dataset at the log scale across time: **log(value) =f(timeidx)**.

Modeling in R

R Function for modeling:

gls() in the package of {nlme}

Important parameters of gls(): (Assuming 6 B-splines used)

- **Model formula**
  - ***model***: log(value) ~ [S1(timeidx)+…+S6(timeidx)]*TRT
- **Parameters to define the covariance matrix**
  - ***weights*** (i.e. the variable function):
    "homoscedastic" or "heteroscedastic“
  - ***correlation*** (i.e. correlation structure):
    CompSymm, AR(1), unstructured, etc.

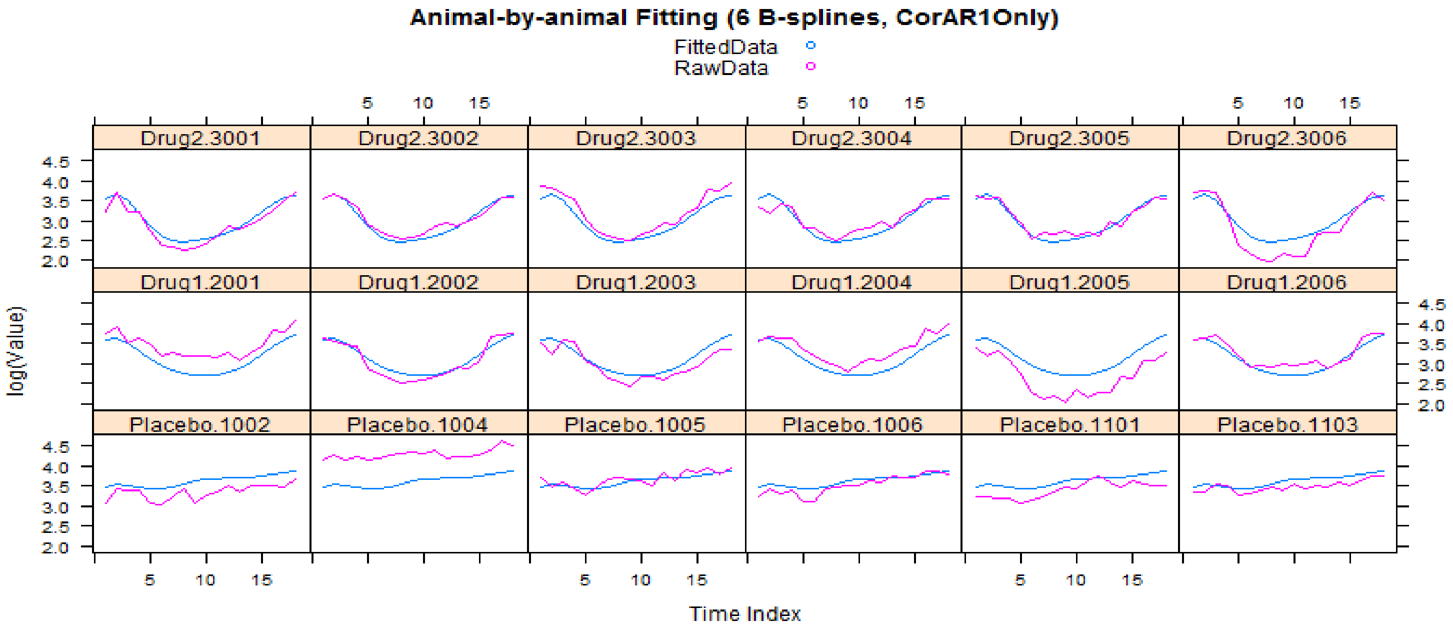
Modeling in SAS

Proc used for modeling: *proc mixed*

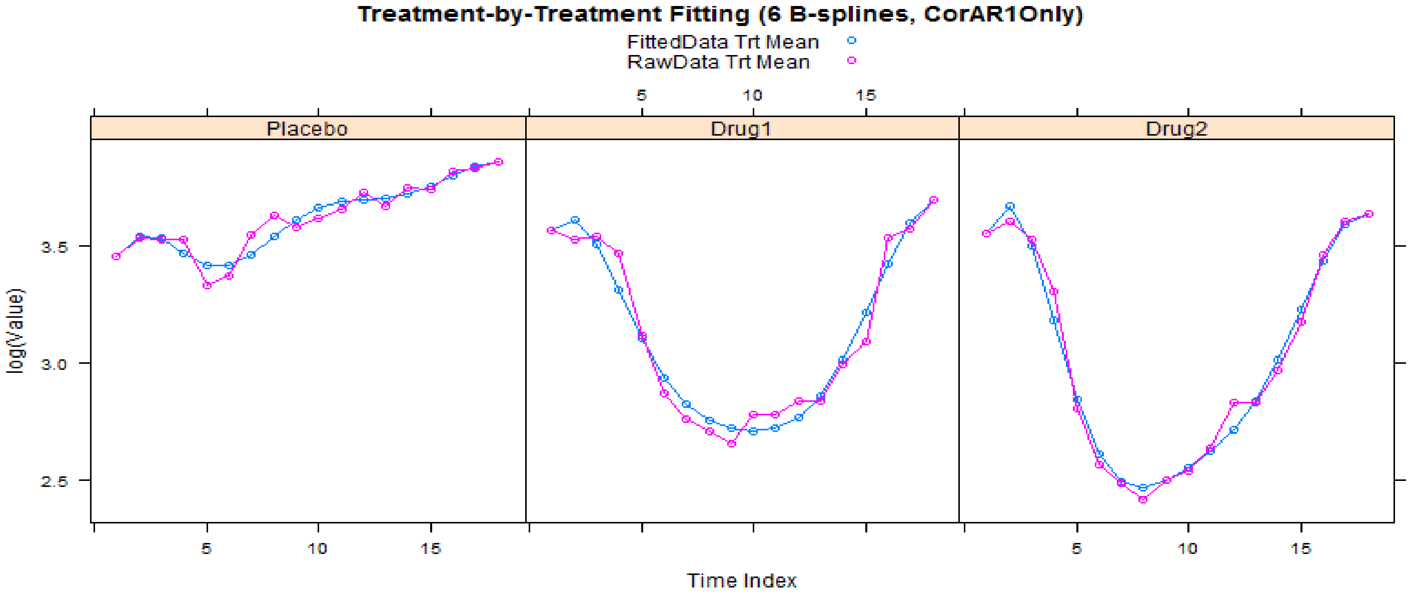
```
proc mixed data=thedata method=REML MMEQ;
          class animalID TRT;
          model value=TRT BF1 BF2 BF3 BF4 BF5 BF6 BF1*TRT
BF2*TRT BF3*TRT BF4*TRT BF5*TRT BF6*TRT / solution
outp=testreml;
          repeated  / type=ar(1) subject=animalID r;
run;
```

Note: BF1 BF2 BF3 BF4 BF5 BF6: 6 B-splines used as bases

Fitted Model When Σ = CorAR1Only(AR(1))

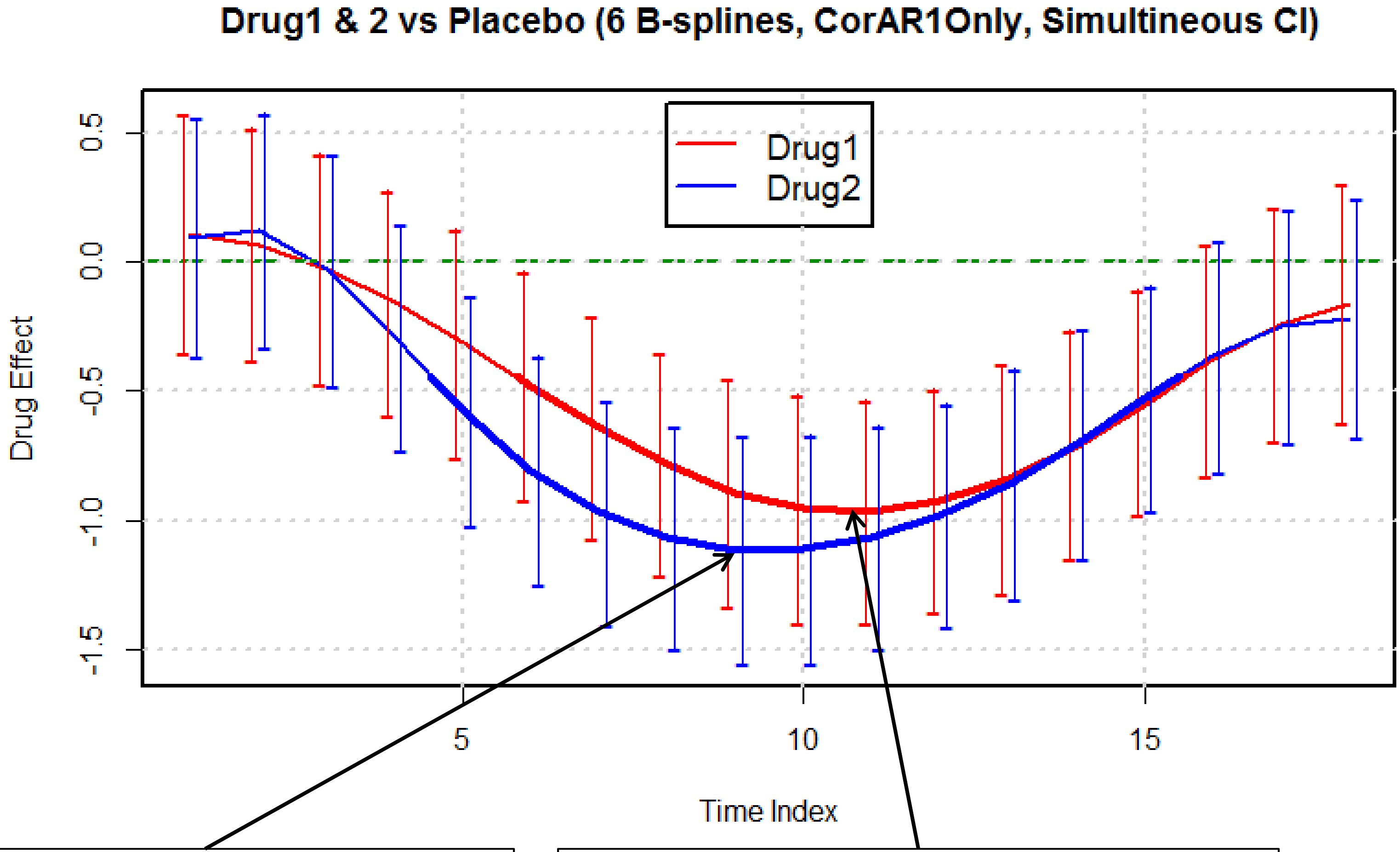


Animal-by-animal Fitting (6 B-splines, CorAR1Only)



Treatment-by-Treatment Fitting (6 B-splines, CorAR1Only)

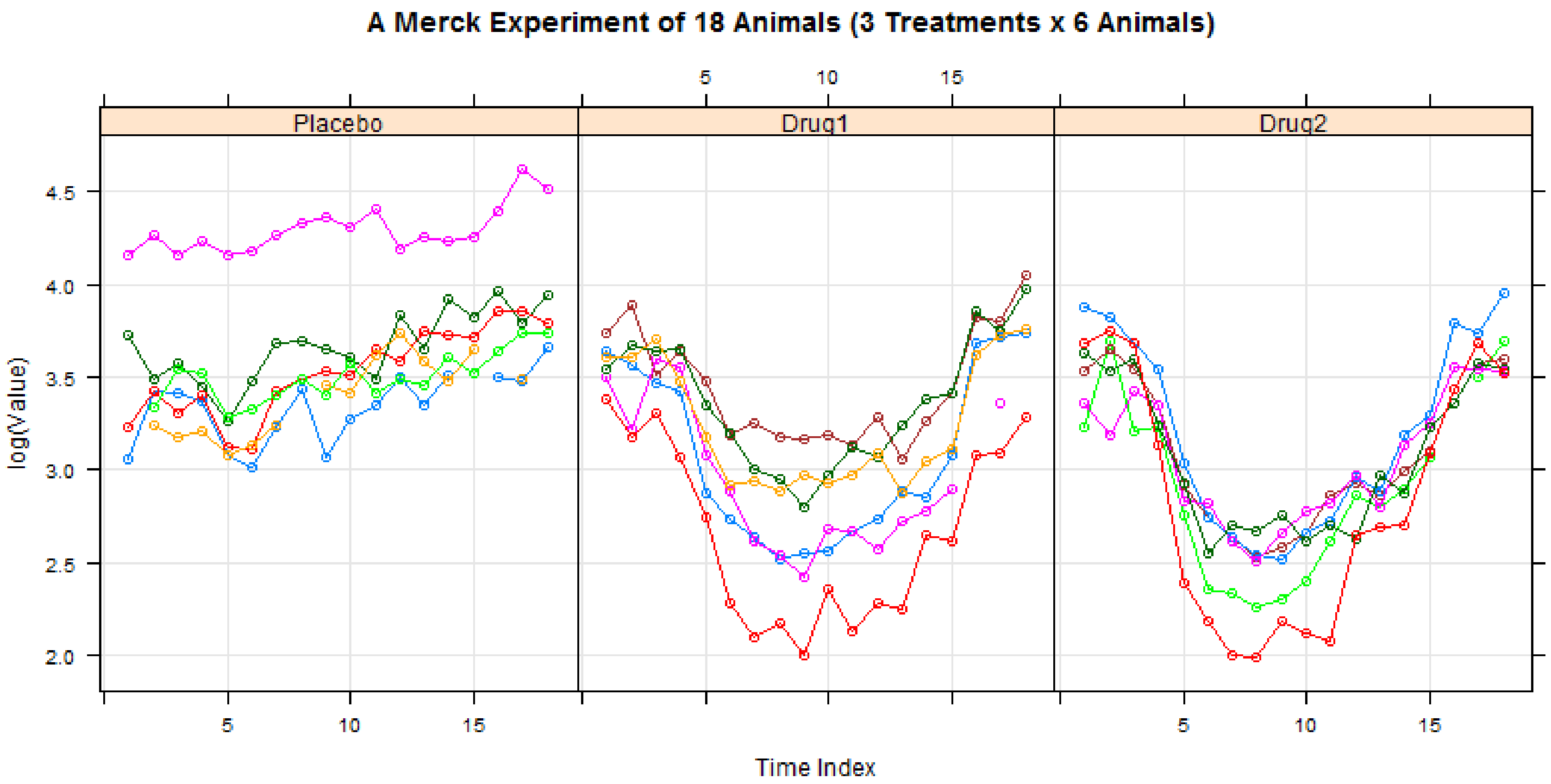
Model Inference When Σ = CorAR1Only



Drug2/Placebo = 0.301, t_max at ~ 9 (time unit)

Drug1/Placebo = 0.375, t_max at ~ 11 (time unit)

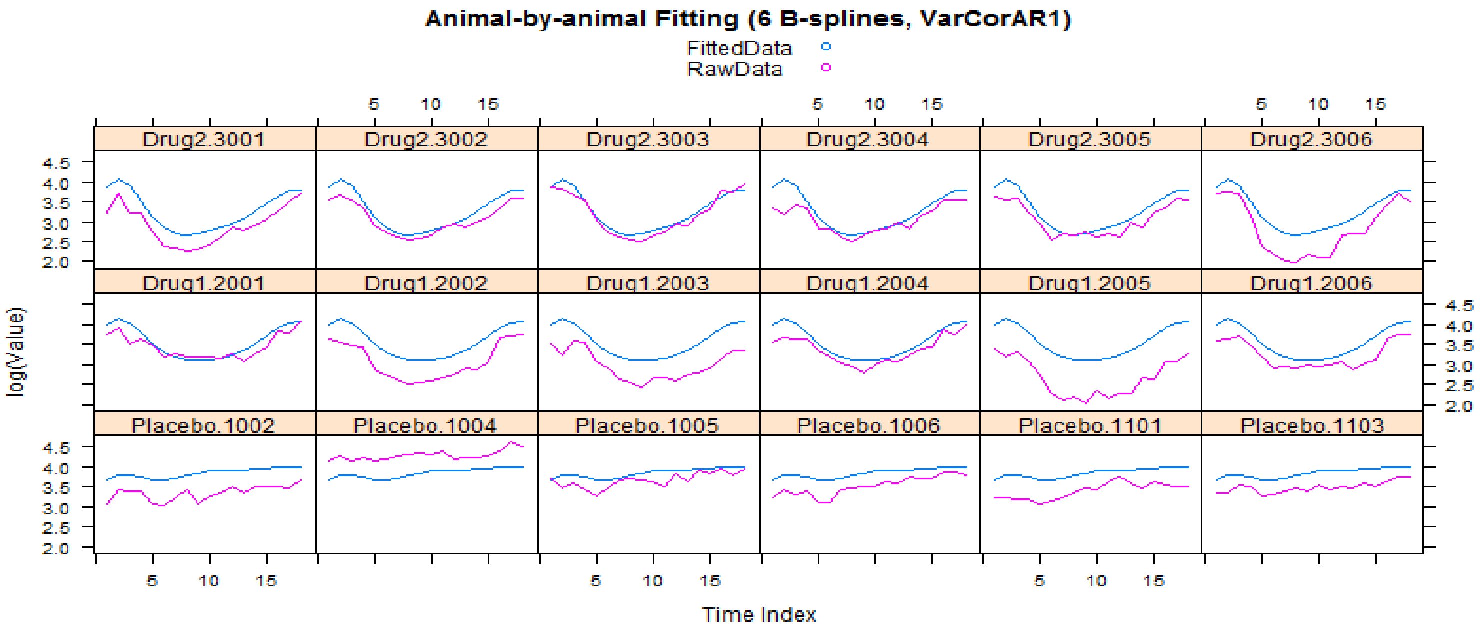
The Example Dataset

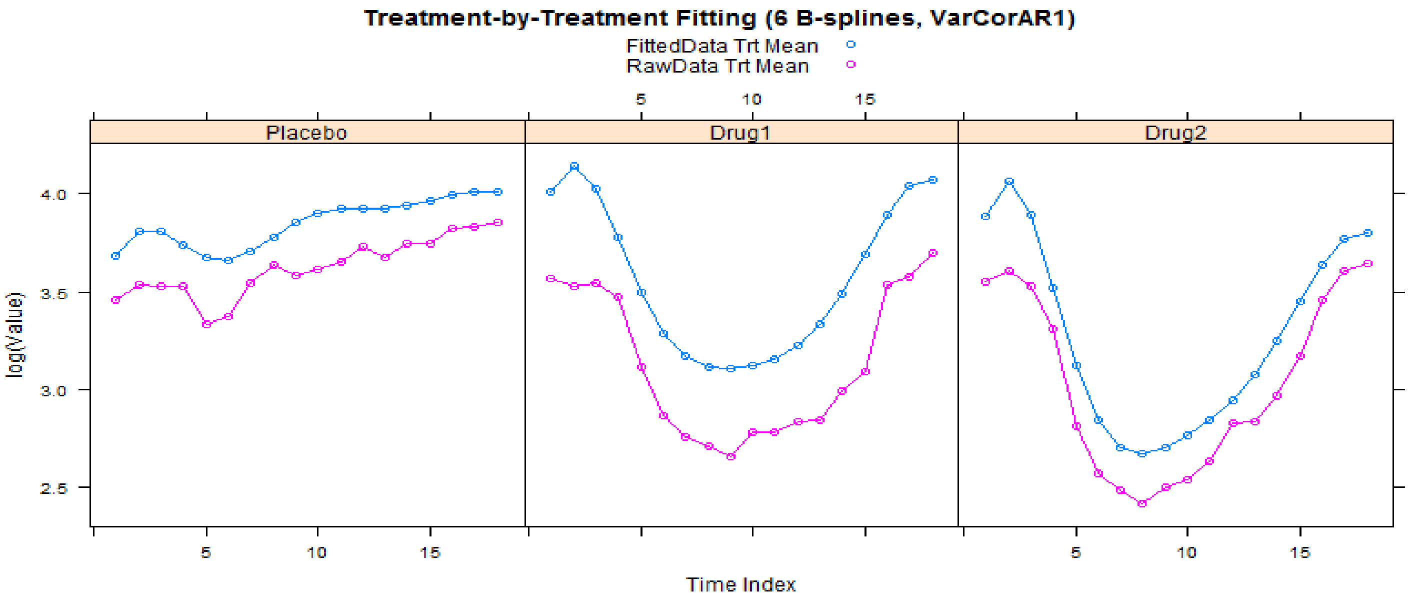


Data suggest a heteroscedastic covariance structure, e.g.VarCorAR1 or VarCorCompSymm (i.e. type=ARH or CSH)

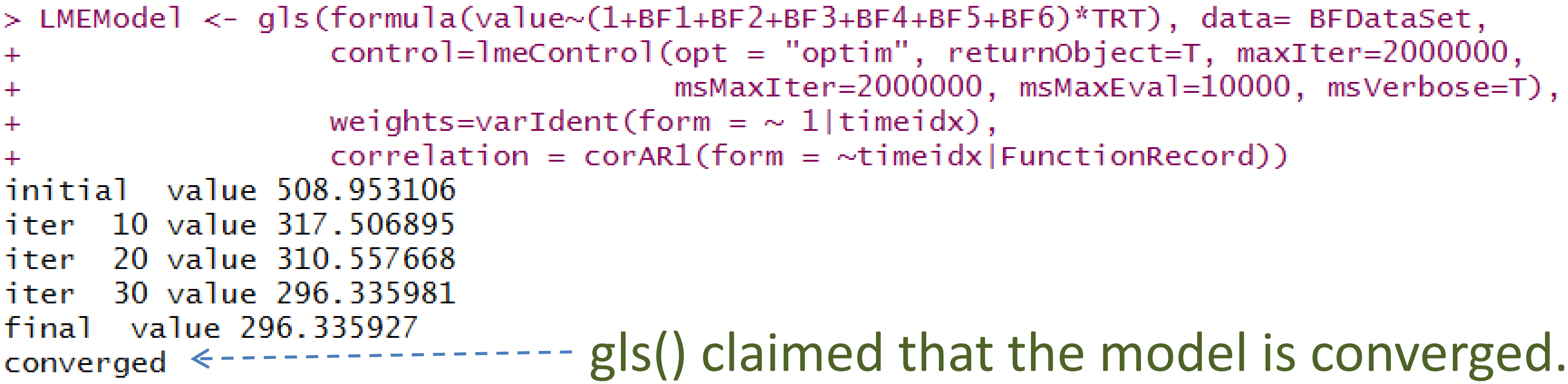
$$\begin{bmatrix} \sigma^2{}_1 & \cdots & \sigma_1\sigma_k\rho^{k-1} \\ \vdots & \ddots & \vdots \\ \sigma_k\sigma_1\rho^{k-1} & \cdots & \sigma^2{}_k \end{bmatrix}$$

Fitted Model When Σ = VarCorAR1


Animal-by-animal Fitting (6 B-splines, VarCorAR1)

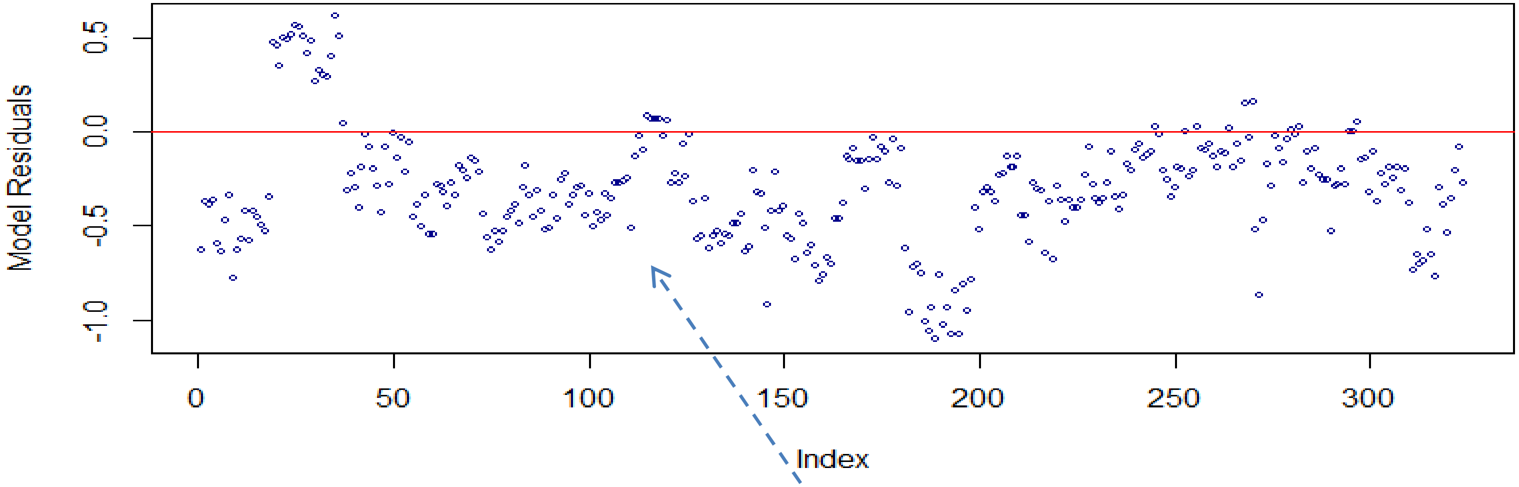

Treatment-by-Treatment Fitting (6 B-splines, VarCorAR1)

Converged to a Bad Model



```
> LMEModel <- gls(formula(value~(1+BF1+BF2+BF3+BF4+BF5+BF6)*TRT), data= BFDataSet,
+                  control=lmeControl(opt = "optim", returnObject=T, maxIter=2000000,
+                                     msMaxIter=2000000, msMaxEval=10000, msVerbose=T),
+                  weights=varIdent(form = ~ 1|timeidx),
+                  correlation = corAR1(form = ~timeidx|FunctionRecord))
initial  value 508.953106
iter  10 value 317.506895
iter  20 value 310.557668
iter  30 value 296.335981
final  value 296.335927
converged
```

gls() claimed that the model is converged.



The residual plot shows that the model does not fit the data.

Outlines

- The bias problem
- **A Bayesian solution**
- A Frequentist’s solution
- Summary

A Bayesian Solution

- **Bayesian** == a philosophy/theory + a modeling approach + a set of numeric solutions.
- Bayesian-based (generalized) linear models are well studied. Under conventional settings, the results are easy to obtain, and reproducible.
- Bayesian software packages encapsulating the technical details are abundantly available.

  Examples: OpenBUGS, WinBUGS, JAGS, **MCMCglmm**, INLA, etc.

A Multivariate Normal Generative Model

Model

$$\mathbf{y}_i \sim N(\mathbf{x}_i\boldsymbol{\beta}, \boldsymbol{\Lambda}) \quad \text{where} \quad i = 1, \quad \cdots, \quad M$$

$$\boldsymbol{\beta} \sim N(\boldsymbol{\beta}_0, \boldsymbol{\Lambda}_\beta)$$

$$\boldsymbol{\Lambda} \sim \text{Inv-Wishart}(\mathbf{V}, \nu)$$

Priors

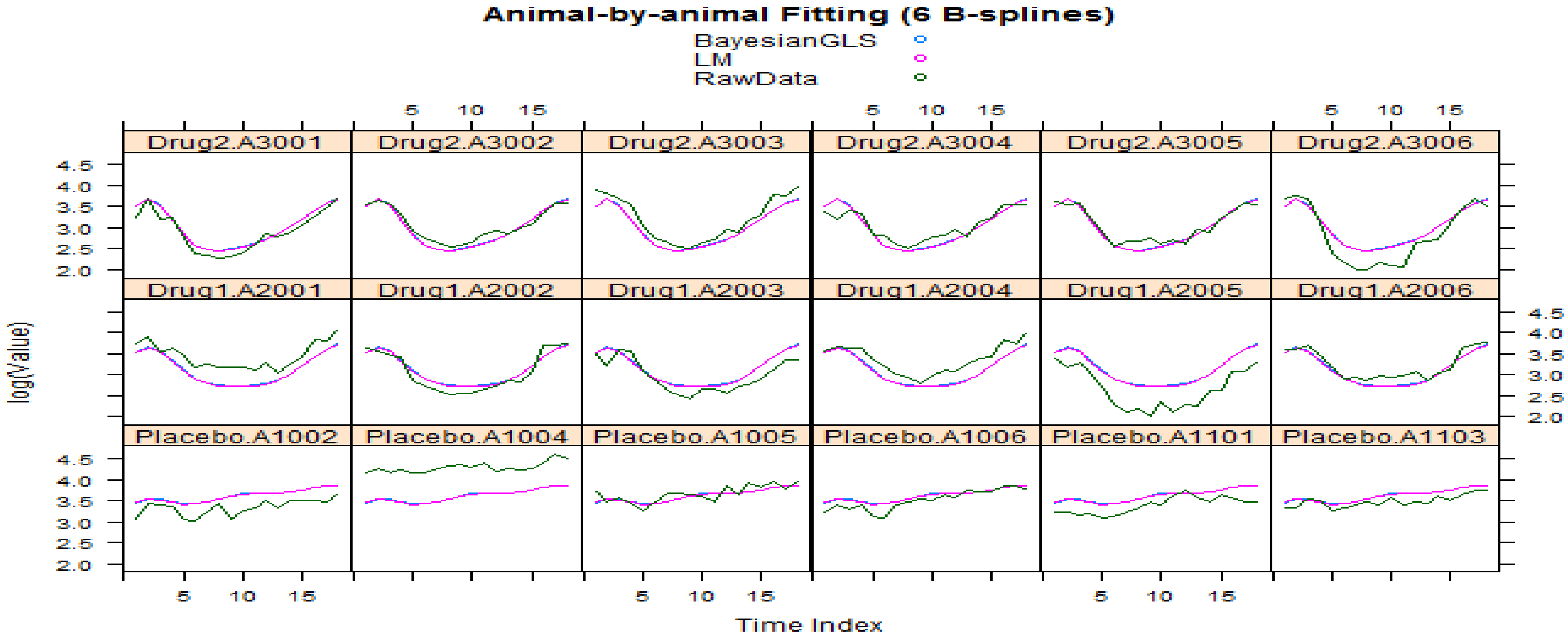
$$\boldsymbol{\beta}_0 = \mathbf{0}_{k\times1}$$

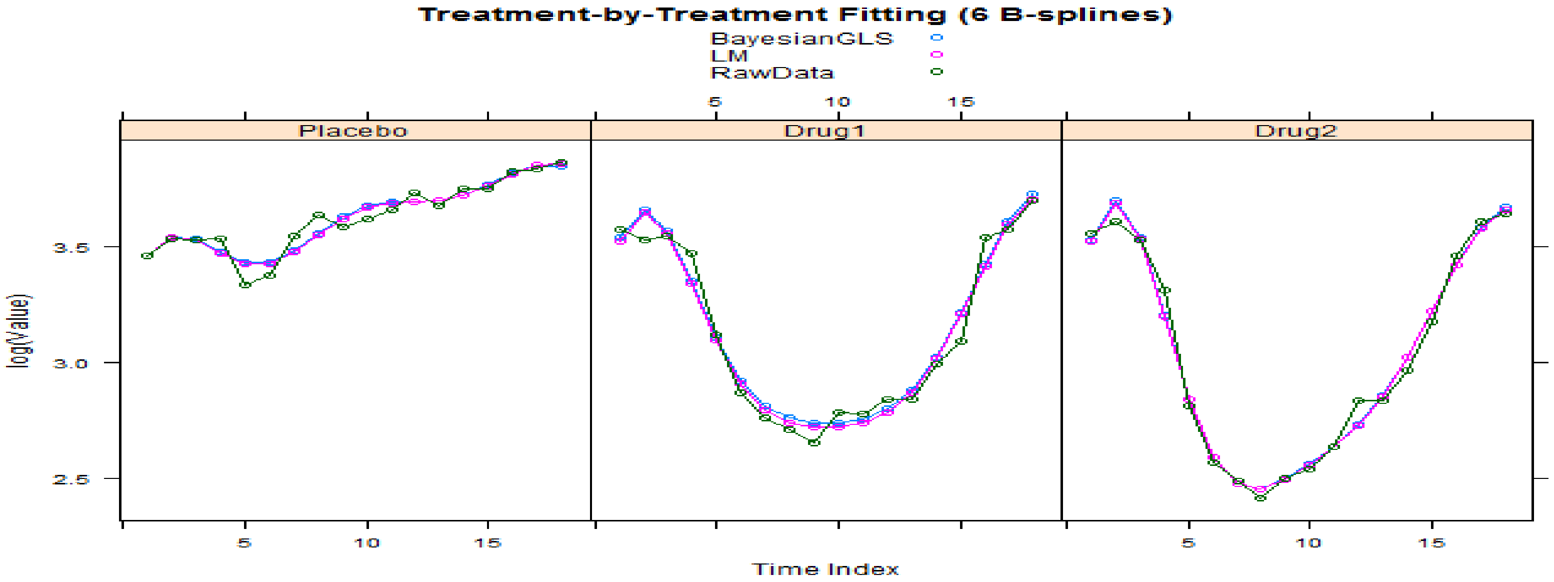
$$\boldsymbol{\Lambda}_\beta = \mathbf{I}_{k\times k} \times 10^{10}$$

$\nu = 2$ ~ prior equivalent sample size

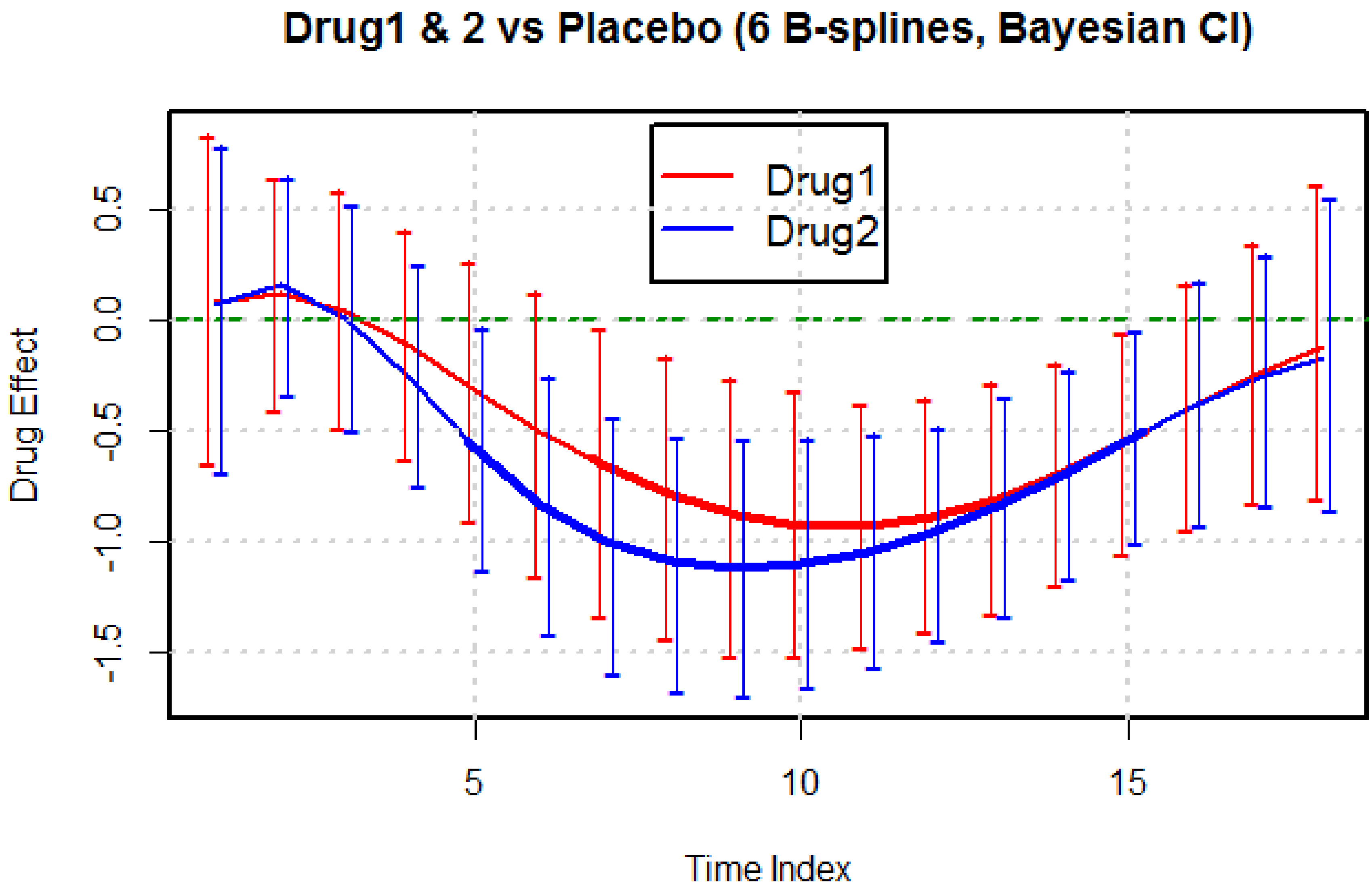
$$\mathbf{V} = \mathbf{I}_{d\times d} + \lambda\mathbf{OD} \qquad \mathbf{OD} = \begin{bmatrix} 0 & 1 & \cdots & 1 \\ 1 & 0 & 1 & \vdots \\ \vdots & 1 & \ddots & 1 \\ 1 & \cdots & 1 & 0 \end{bmatrix}_{d\times d} \qquad \lambda = 0$$

The Obtained Model





The Result



Drug1 & 2 vs Placebo (6 B-splines, Bayesian CI)
Drug1
Drug2
Drug Effect
0.5
0.0
-0.5
-1.0
-1.5
5
10
15
Time Index

Outlines

- The bias problem
- A Bayesian solution
- **A Frequentist's solution**
- Summary

A Frequentist's Solution

- **Frequentists**: We must provide a solution!
- **Basic Idea**: directly optimizing the ***full likelihood***, instead of the ***profile likelihood***.
- **Key issue**: properly estimating **covariance matrix Σ** with complex structures.

Covariance $\mathbf{\Sigma}$ Estimation via Decomposition

- The **covariance matrix** $\mathbf{\Sigma}$ is hard to be directly estimated because of its **symmetry** and **positive-definiteness**.

- **Covariance Matrix Decomposition:** $\mathbf{\Sigma} = \mathbf{BMB'}$
  - Variance-Correlation Decomposition

    B – A diagonal matrix of STD;

    M – The correlation matrix.
  - Spectral Decomposition

    B – An orthogonal matrix of normalized eigenvectors;

    M – A diagonal matrix of eigenvalues.
  - **Cholesky Decomposition**

    B – A lower-triangle matrix with 1 as diagonal elements;

    M – A diagonal matrix

Covariance Matrix Cholesky Decomposition

- **Dr. Mohsen Pourahmadi** pioneered using Cholesky decomposition in covariance matrix estimation since late 1990s. *(Pourahmadi M, 1999; Pourahmadi M 2000.)*

- **Cholesky decomposition:** $\mathbf{\Sigma} = \mathbf{CC'} = \mathbf{LDL'}$

  C — A lower-triangle matrix

  $\mathbf{D} = diag(\mathbf{C}) * diag(\mathbf{C})$

  $\mathbf{L} = \mathbf{CD}^{-1/2}$

- **Modified Cholesky decomposition:** $\mathbf{T\Sigma T'} = \mathbf{D}$

  The key attraction is its connection with the **linear auto-regression model**.

Covariance Matrix vs. Auto-regression

- Data $\mathbf{Y} = [Y_1 \quad \cdots \quad Y_n]'$ and Covariance $\mathbf{\Sigma} = \mathrm{cov}(\mathbf{Y})$
- Modified **Cholesky decomposition:** $\mathbf{T\Sigma T}' = \mathbf{D}$
- **Linear Auto-regression:** $Y_t = \sum_{j=1}^{t-1} \phi_{t,j} Y_j + \varepsilon_t \qquad t = 1 \quad \cdots \quad n$
- **Connections:** *Covariance Estimation = Regression Estimation*

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -\phi_{2,1} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -\phi_{n,1} & -\phi_{n,2} & \cdots & 1 \end{bmatrix}$$

-- dependence

$$\mathbf{D} = \mathrm{cov}([\varepsilon_1 \quad \cdots \quad \varepsilon_n]') = \begin{bmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_n^2 \end{bmatrix}$$

-- variance

Covariance Matrix Parameterization

- $\boldsymbol{\phi_{tj}}, \boldsymbol{\sigma_t}$ can be parameterized in linear functions:
  - dependence parameters: $\phi_{tj} = \mathbf{z}_{tj}^{'}\boldsymbol{\gamma}$
  - variance parameters: $\log \sigma_t^2 = \mathbf{e}_t^{'}\boldsymbol{\lambda}$

  where the pre-specified $\boldsymbol{z_{tj}}, \boldsymbol{e_t}$ are $q \times 1$ and $d \times 1$ vectors.

- Several choices (*e.g. polynomial, AR(1), step function*) are available for the design vectors $\boldsymbol{z_{tj}}, \boldsymbol{e_t}$. They generally produce good results.

- Covariance estimation: $\boldsymbol{\Sigma}_{\mathbf{z,e}}^{-1}(\gamma, \lambda) = \mathbf{T_z}(\gamma)\mathbf{D_e}(\lambda)^{-1}\mathbf{T_z}(\gamma)'$

Regression Model Re-parameterization

- The original regression model

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \qquad \boldsymbol{\varepsilon} \sim N(0, \boldsymbol{\Sigma})$$

can be re-parameterized as a ***dynamic*** linear model

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}_{\mathbf{Y}}\boldsymbol{\gamma} + \mathbf{d} \qquad \mathbf{d} \sim N(0, \mathbf{D}(\boldsymbol{\lambda}))$$

where $\mathbf{Z}_{\mathbf{Y}} = [\mathbf{Z}_{\mathbf{Y}}[1] \quad \cdots \quad \mathbf{Z}_{\mathbf{Y}}[n]]'$

$$\mathbf{Z}_{\mathbf{Y}}[t] = \sum_{j=1}^{t-1} \mathbf{R}_{\mathbf{Y}}[j]\mathbf{z}_{tj} \qquad \mathbf{R}_{\mathbf{Y}} = \mathbf{Y} - \mathbf{X}\boldsymbol{\beta}$$

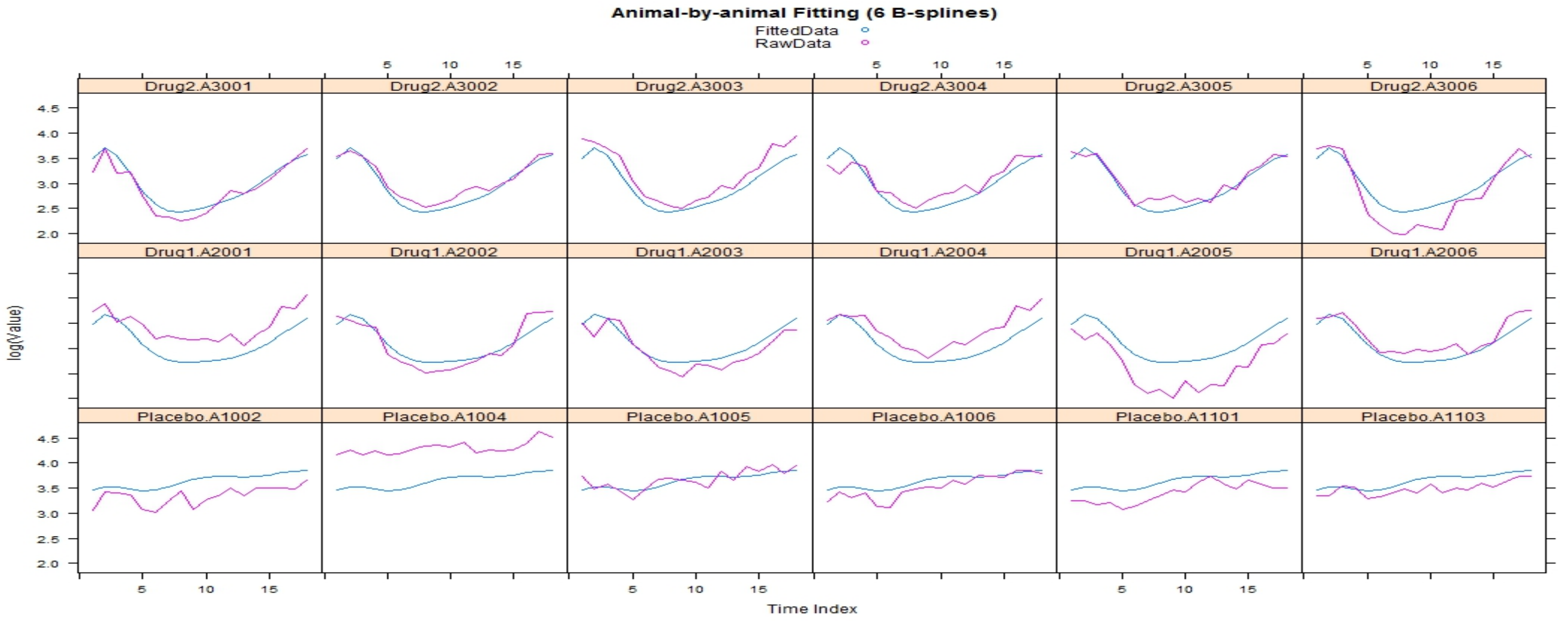
- Model parameters are changed from {**β,Σ**} to {**β,γ,λ**}

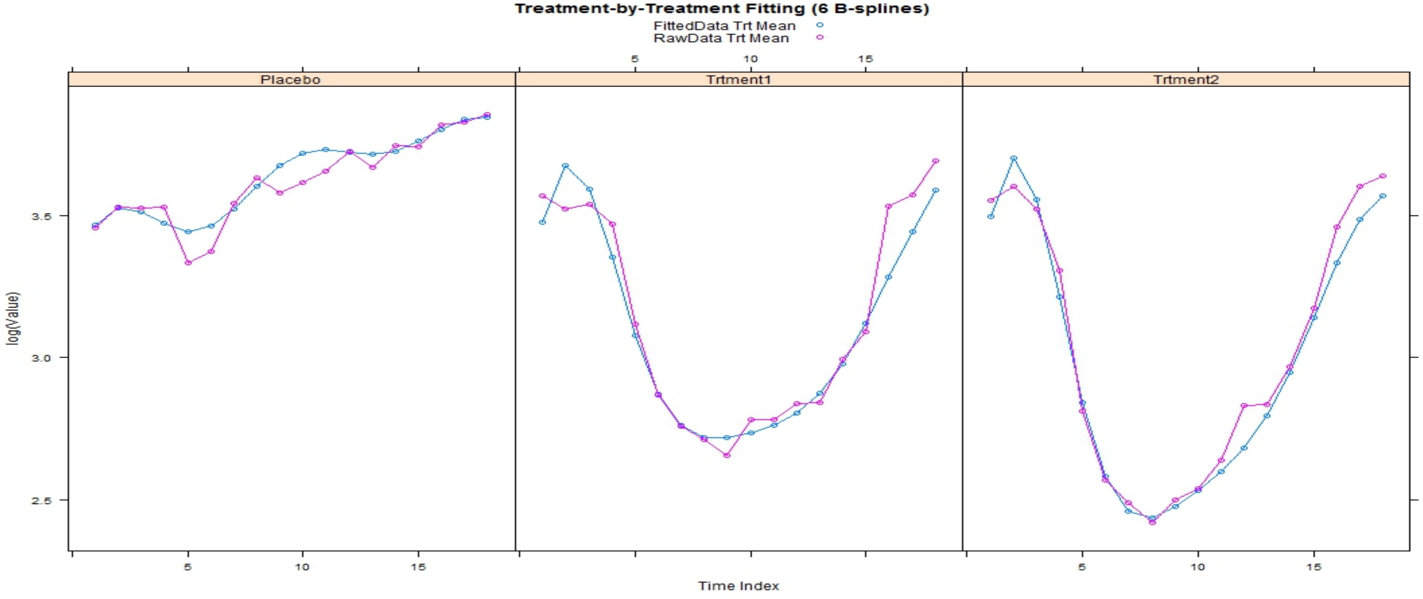
An Iteratively Reweighted LS Algorithm *

1. Determine {**β**,**γ**}'s initial values as {$\boldsymbol{\beta}_0$,$\boldsymbol{\gamma}_0$}.
2. Estimate the variance parameter **λ** based on the residuals of $\mathbf{R2}_i = \mathbf{Y}_i - (\mathbf{X}_i\boldsymbol{\beta}_0 + \mathbf{Z}_{\mathbf{Y}_i}\boldsymbol{\gamma}_0) \quad i = 1 \cdots m$ **
3. Obtain **T** and **D** via calculating $\phi_{tj} = \mathbf{z}_{tj}^{'}\boldsymbol{\gamma}_0$ and $\sigma_t^2 = \exp(\mathbf{e}_t^{'}\boldsymbol{\lambda})$
4. Compute covariance matrix as $\boldsymbol{\Sigma}^{-1} = \mathbf{T}\mathbf{D}^{-1}\mathbf{T}'$
5. Update $\boldsymbol{\beta} = (\sum_{i=1}^{m} \mathbf{X}_i^{'}\boldsymbol{\Sigma}^{-1}\mathbf{X}_i)^{-1}\sum_{i=1}^{m} \mathbf{X}_i^{'}\boldsymbol{\Sigma}^{-1}\mathbf{Y}_i$
6. Compute residuals $\mathbf{R}1_i = \mathbf{Y}_i - \mathbf{X}_i\boldsymbol{\beta} \quad i = 1 \cdots m$
7. Update $\mathbf{Z}_{\mathbf{Y}_i}$ based on $\mathbf{R}1_i$ as $\mathbf{Z}_{\mathbf{Y}}[t] = \sum_{j=1}^{t-1} \mathbf{R1}_i[j]\mathbf{z}_{tj}$
8. Update $\boldsymbol{\gamma} = (\sum_{i=1}^{m} \mathbf{Z}_{\mathbf{Y}i}^{'}\boldsymbol{\Sigma}^{-1}\mathbf{Z}_{\mathbf{Y}i})^{-1}\sum_{i=1}^{m} \mathbf{Z}_{\mathbf{Y}i}^{'}\boldsymbol{\Sigma}^{-1}\mathbf{R1}_i$
9. Update $\mathbf{R2}_i = \mathbf{R1}_i - \mathbf{Z}_{\mathbf{Y}_i}\boldsymbol{\gamma}$
10. Stop the process if {$\boldsymbol{\beta}_0$,$\boldsymbol{\gamma}_0$}~= {**β**,**γ**}, otherwise {$\boldsymbol{\beta}_0$,$\boldsymbol{\gamma}_0$} = {**β**,**γ**}, and go to step 2.

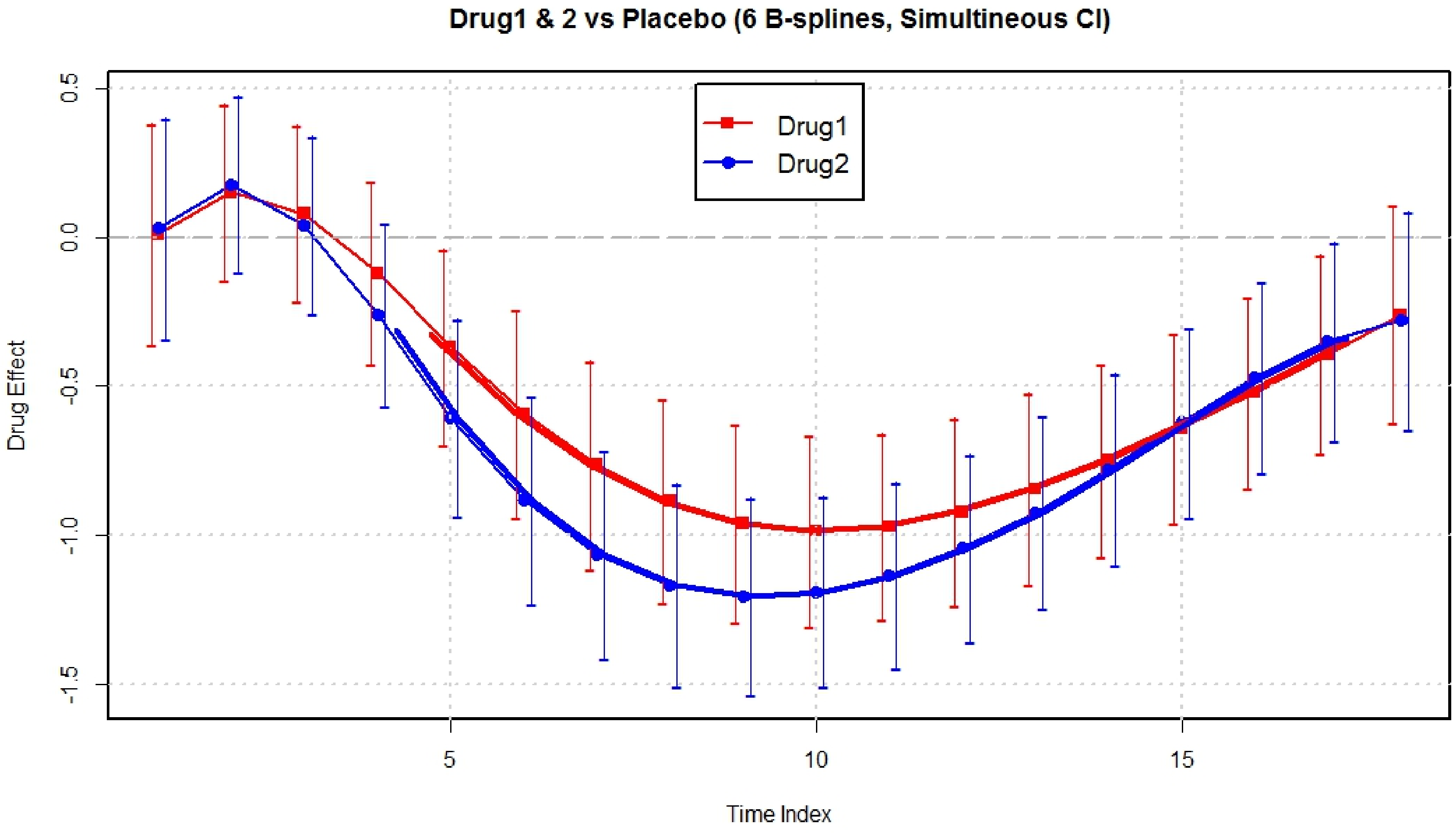
Note: Used ~400 lines of R codes to implement this algorithm.
*References: * Daniels MJ and Pourahmadi M, 2002. ** Verbyla AP, 1993.*

Fitted Model



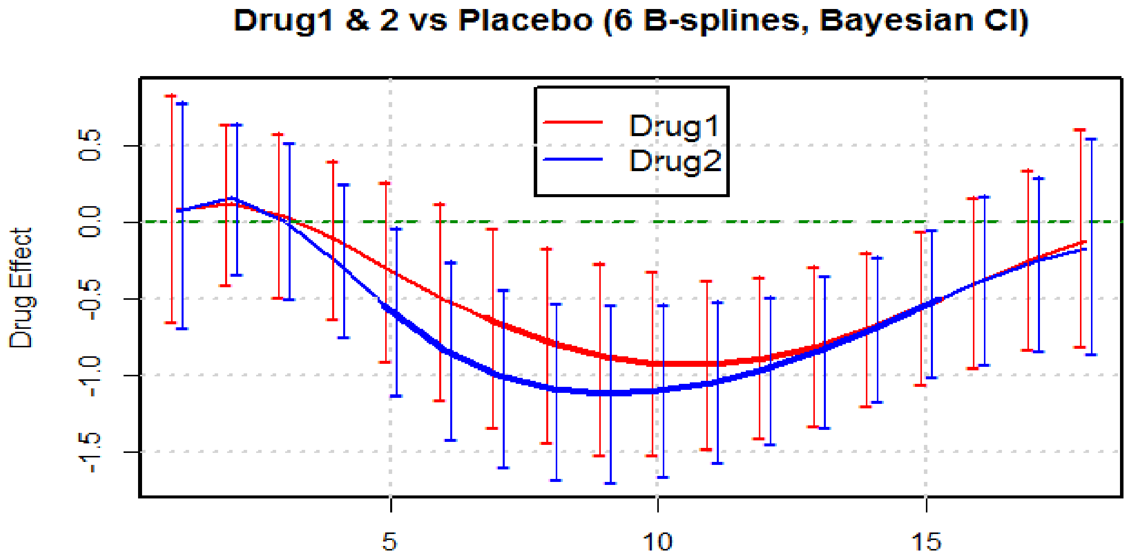


The Result

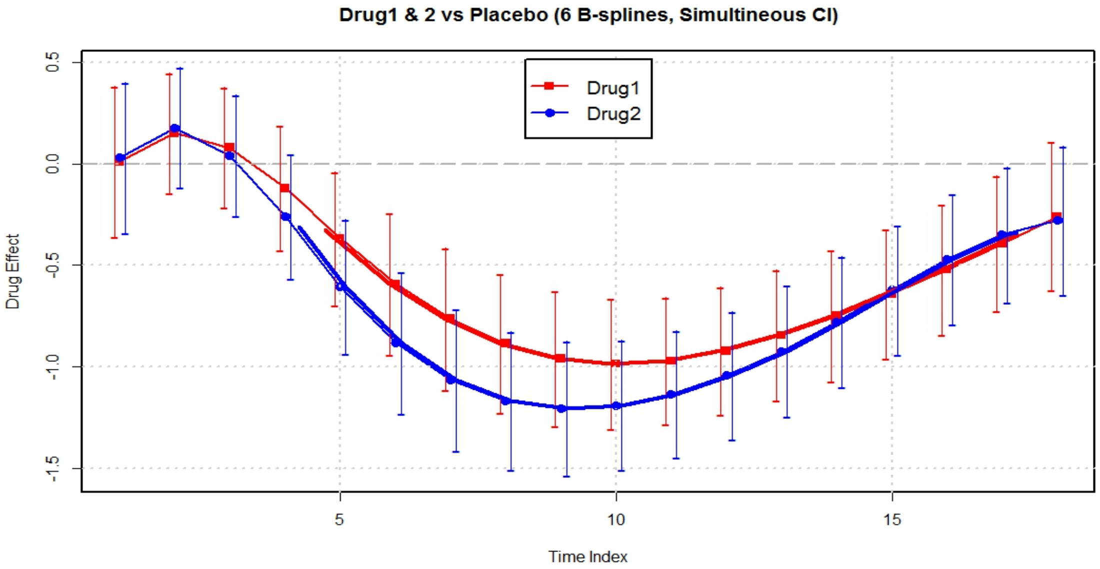

Drug1 & 2 vs Placebo (6 B-splines, Simultineous CI)
Drug1
Drug2
Drug Effect
0.5
0.0
-0.5
-1.0
-1.5
5
10
15
Time Index

The Results

Bayesian Method



Frequentist's Method



Outlines

- The bias problem
- A Bayesian solution
- A Frequentist’s solution
- **Summary**

Summary

- Large biases in fixed effects caused by failed optimization in popular statistical packages, including R (NLME) and SAS (Proc Mixed).
- Problem: Pinpoint the exact place in the optimization where this issue happens.
- The issue can be circumvented/reduced using either a Bayesian or a Frequentist’s approach.
- As for algorithmic coding, the Bayesian method is easier. As for future usage, the frequentist-based method is more straightforward (at least to me).