Overcoming long run time for Bayesian integrated fish population models

AppliBugs, 10 June 2020

Etienne Rivot

ESE Ecology and Ecosystem Health
Rennes, France
etienne.rivot@agrocampus-ouest.fr

Atelier Modèles Hiérarchiques

HBM - MCMC - BUGS & Fisheries

I love you, neither !

Stock assessment models

- **Population dynamics - Life cycle models**
  - **High dimensionality : age, stage, time, space …**
  - **Non linear**
  - **Highly stochastic**

- **Hierarchical statistical structure**

  Latent states

  Multiple sources of data (integrated models)

Aeberhard, W. H., Mills Flemming, J., & Nielsen, A. (2018). Review of State-Space Models for Fisheries Science. Annual Review of Statistics and Its Application, 5(1), 215-235. https://doi.org/10.1146/annurev-statistics-031017-100427

Long run time may be a serious bottleneck

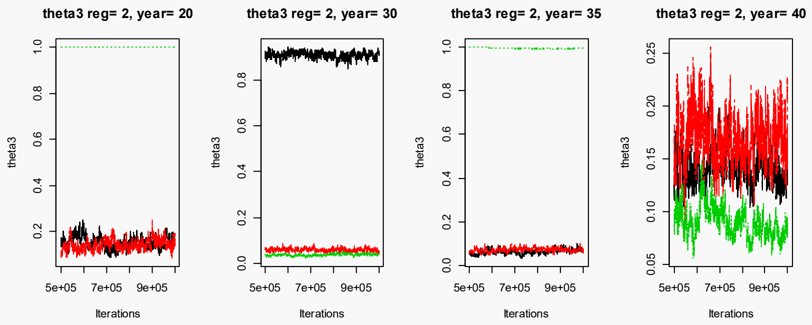
- Bayesian methods are advocated for fisheries stock assessment

  Hierarchical / State-space models **/** Informative priors **/** Risk analysis

  Punt, A. E., & Hilborn, R. (1997). Fisheries stock assessment and decision analysis : The Bayesian approach. *Reviews in Fish Biology and Fisheries*, *7*, 35-63.

- But are still rarely used in practice (e.g., working groups) because of prohibitive run time (~ of the order of days to months)

  Difficult to explore model sensitivity and to evaluate different options during model development or the review process



Long run time may be a serious bottleneck

Monnahan, C.C., Branch, T. A., Thorson, J.T., Stewart, I.J., & Szuwalski, C.S. (2019). Overcoming long Bayesian run times in integrated fisheries stock assessments. ICES Journal of Marine Science. https://doi.org/10.1093/icesjms/fsz059

Table 1. Summary of case studies used.

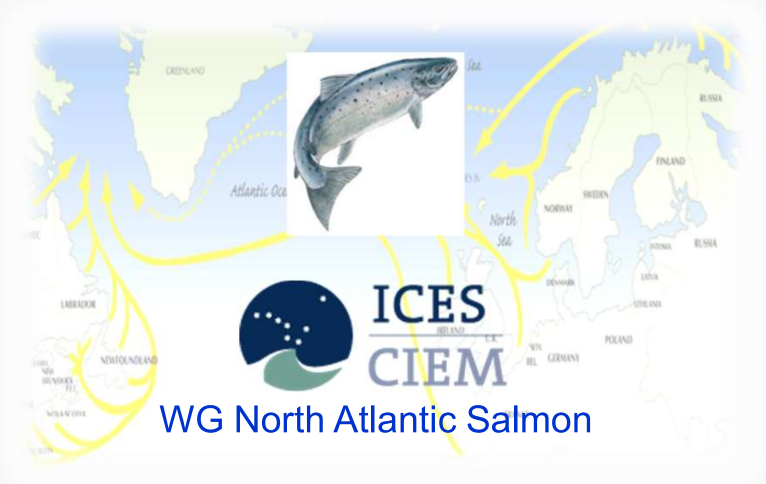
Model name	No. of parameters	Speed (s 1000^{-1} evals)	Brief description	Species and reference	Time needed for ESS = 1000
Hake	217	8.71	MCMC results used for management, empirical weight-at-age, Stock Synthesis	Pacific hake; *Merluccius productus* (Grandin *et al.*, 2016)	18 h
Halibut	195	24.06	Time-varying catchability, empirical weight-at-age, Stock Synthesis	Pacific halibut; *Hippoglossus stenolepis* (Stewart *et al.*, 2016)	12 months
Canary	304	188.10	Time-varying growth, three areas with different exploitation history but no movement, natural mortality varies by age for males, complex selectivity with 31 fleets, Stock Synthesis	Canary rockfish; *Sebastes pinniger* (Thorson and Wetzel, 2015)	187 months
Snow crab	334	18.57	Length-structured, custom built, considerations for sex, maturity state, and shell condition, growth per moult data available	Eastern Bering Sea snow crab; *Chionoecetes opilio* (Szuwalski and Turnock, 2016)	38 months

Speed is how many seconds 1000 model evaluations take and is calculated as warmup and sampling time (but not optimization) divided by the total iterations during a RWM runs in which gradients are not calculated.

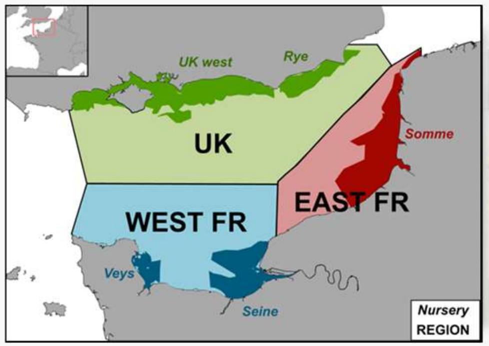
HBM-MCMC-BUGS & E S E

Integrated fish population models

Atlantic salmon in the North Atlantic Ocean (basin scale)



Common Sole in the Eastern Channel



Do HBM-MCMC-BUGS free the modeler ?

Frédéric Gosselin, 2017

Atelier Modèles Hiérarchiques

Do HBM-MCMC-BUGS freeze the modeler ?

Frédéric Gosselin, 2017

Atelier
Modèles
Hiérarchiques

Identifying strategies to improve MCMC performance is becoming increasingly crucial as the complexity of models, and the run times to fit them, increases

Strategies to overcome long run timing

- Forget Bayesian methods
  Use Optimization approaches (max Likelihood)

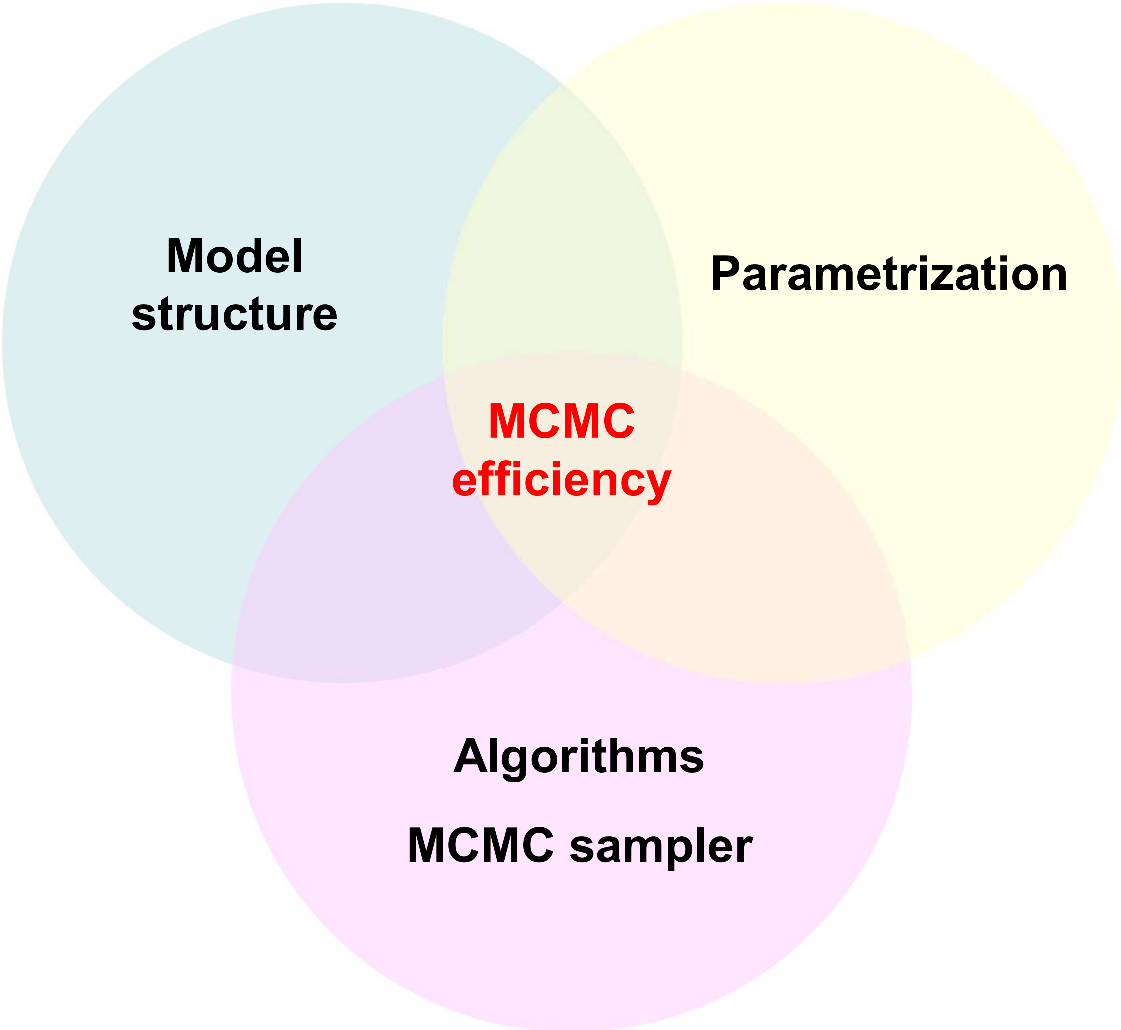
- Simplify the model - Use coarser approximations to the pop. dyn.

- Faster computers
- Run MCMC chains in parallel

- Model structure
- Prior (regularization, informative priors)
- Parameterization
- Sampling strategy

"One size does not fit all"

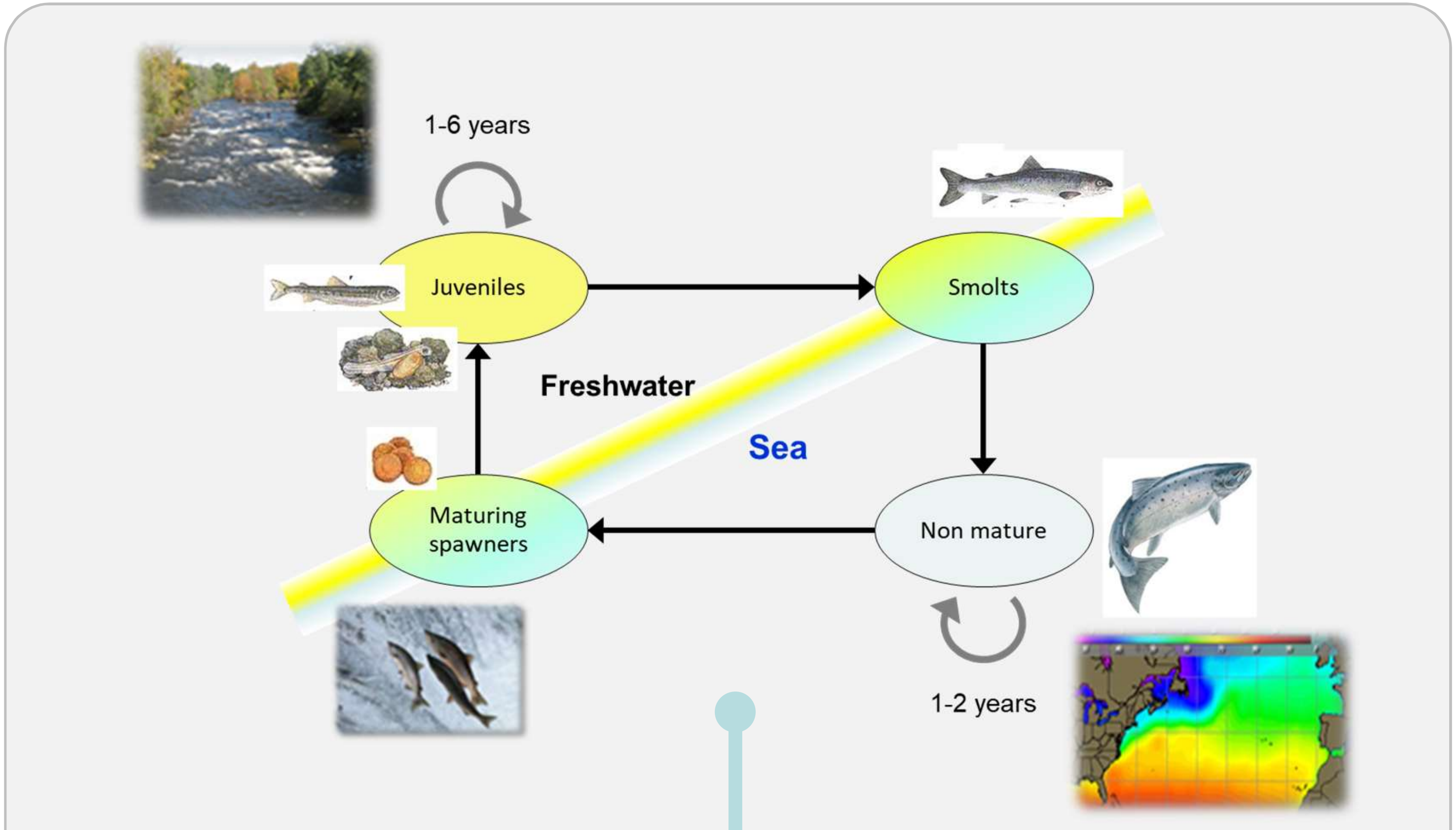
Ponisio, L. C., de Valpine, P., Michaud, N., & Turek, D. (2020). One size does not fit all : Customizing MCMC methods for hierarchical models using NIMBLE. *Ecology and Evolution*, *10*(5), 2385-2416. https://doi.org/10.1002/ece3.6053



Outlines

- Case study - Integrated population model for Atlantic salmon
- Strategies to overcome long run time NIMBLE

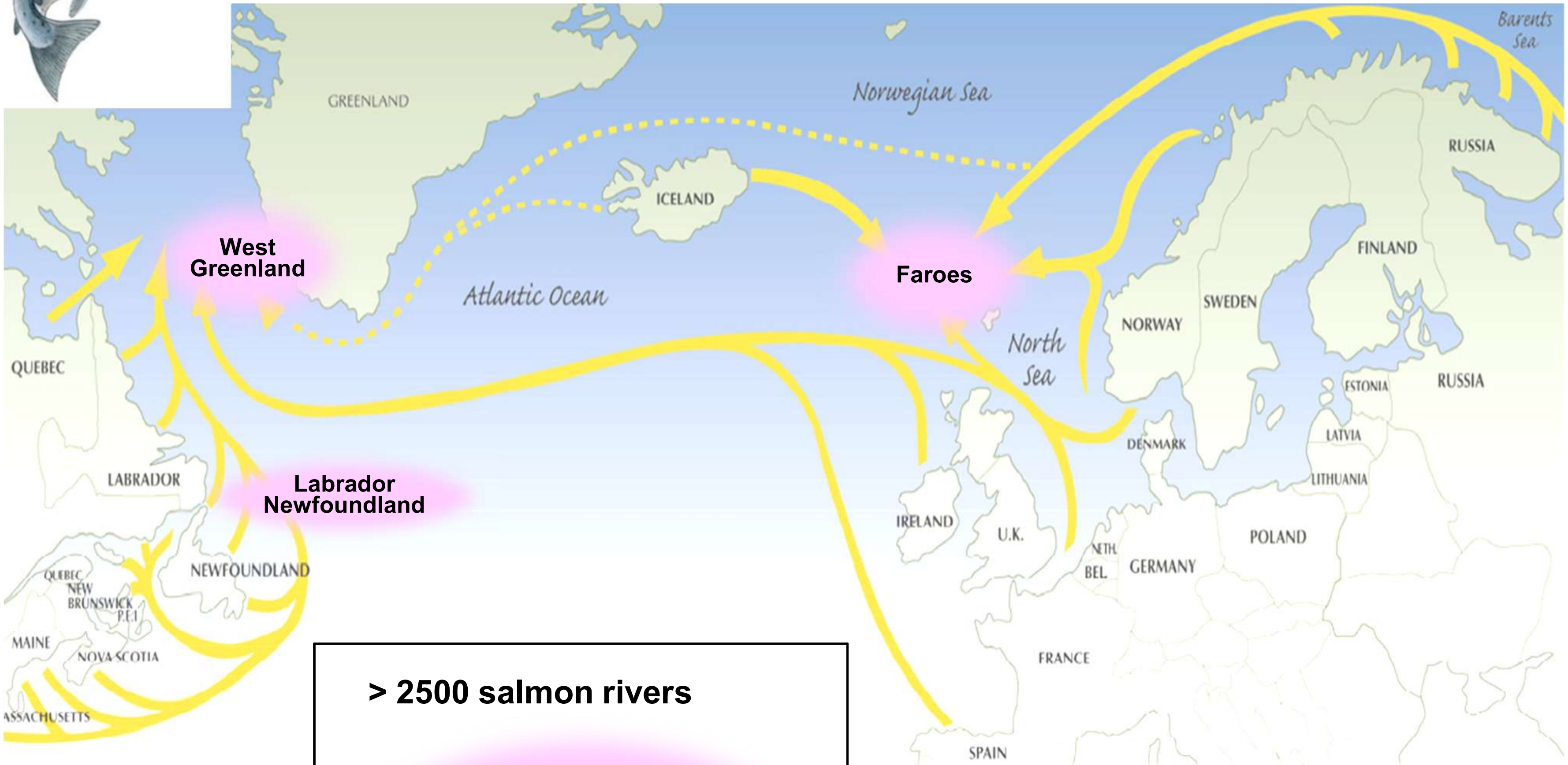
Integrated population model for Atlantic salmon (*salmo salar*)



- **Survival**
- **Prob. to mature after one year spent at sea**

Olmos, M., … & Rivot, E. (2019). *Fish and Fisheries*, *20*(2), 322-342. https://doi.org/10.1111/faf.12345
Olmos, M., … & Rivot, E. (2020). *Global Change Biology*, 26: 131ç-1337. https://doi.org/10.1111/gcb.14913

Migration routes in the North Atlantic ocean

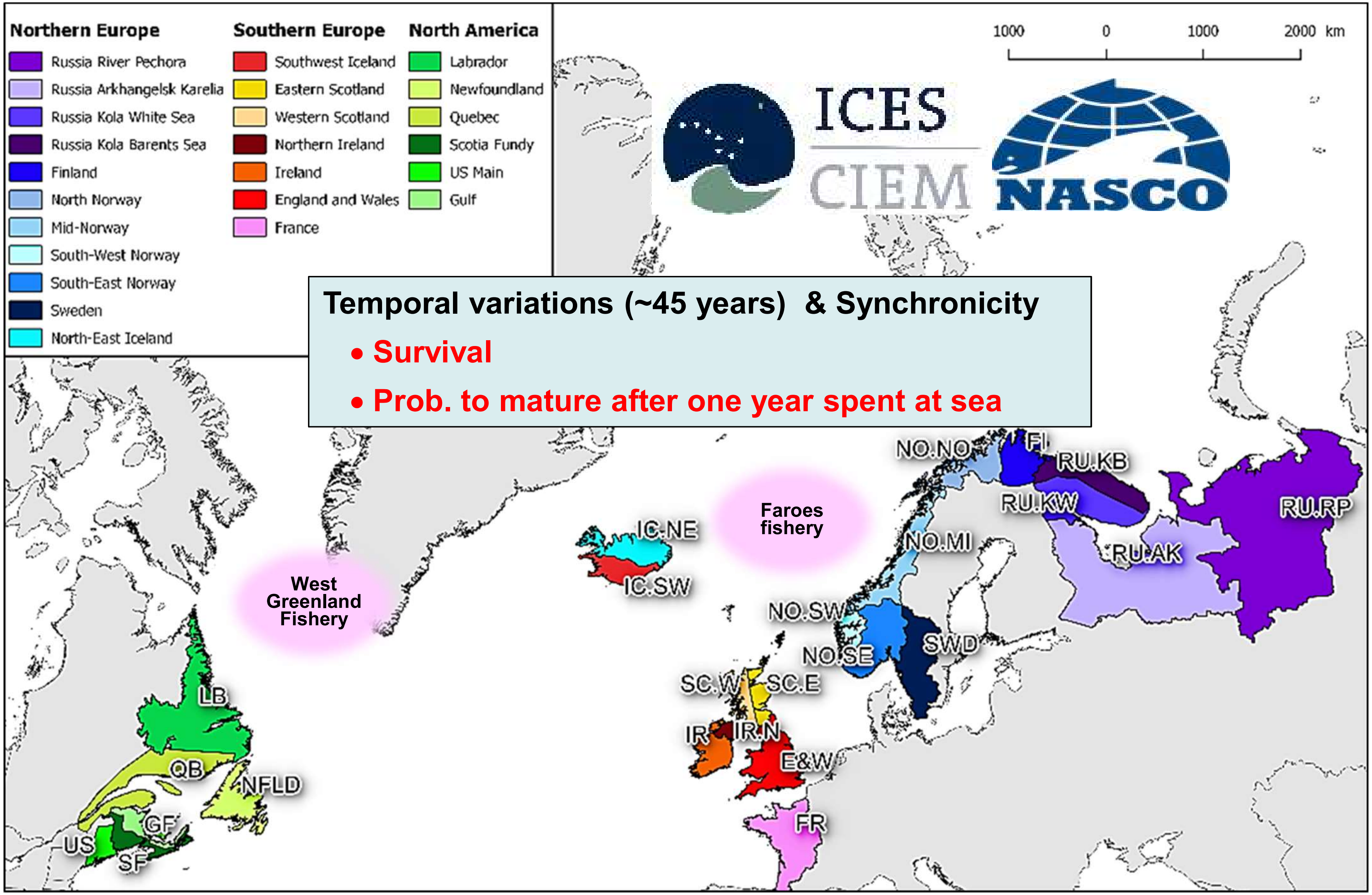


> 2500 salmon rivers

Foraging areas
and
fisheries operating on
mixed stocks

Source : Modified from Atlantic Salmon Federation

24 stock units in the North Atlantic ocean



Temporal variations (~45 years) & Synchronicity

- **Survival**
- **Prob. to mature after one year spent at sea**

Adapted from Olmos et al., Fish and Fisheries, 2019

Core process equations

- **Stochasticity and synchronicity in**

  . Marine survival

  . Proportion of fish maturing after one year spent at sea

$$logit(\theta_{t+\mathbf{1},\mathbf{1}:\boldsymbol{S}}) \sim MultiNorm^{S}\left(logit(\theta_{t,\mathbf{1}:\boldsymbol{S}}), \Sigma\right)$$

Multi-variate random walk
Dimension S = 24

- **Demographic stochasticity**

$$\log(N_{i+1,t+1,s}) \sim N\left(\log(\theta_{i,t,s} \cdot N_{i,t,s}), \sigma\right)$$

σ fixed to low value

Comparing different configurations

NIMBLE

- Benchmarking

- Baseline version (with “good” inits)

- Model structure and parameterization
  - Deterministic transitions
  - Customized distributions to integrate out transitions
  - Prior for variance-covariance matrix

- Playing with block sampling

Comparing different configurations

NIMBLE

- Benchmarking
- Baseline version (with “good” inits)
- Model structure and parameterization
  - Deterministic transitions
  - Customized distributions to integrate out transitions
  - Prior for variance-covariance matrix
- Playing with block sampling

Comparing different configurations

- **Same computational effort**

  - 2 independent chains (parallel cores)
  - burnin before thin = 50000
  - thin = 300
  - posterior samples kept per chain after burnin and after thin = 2000

- **Criteria to measuring MCMC efficiency**

  - Algorithmic efficiency
  - Computational efficiency

Algorithmic efficiency

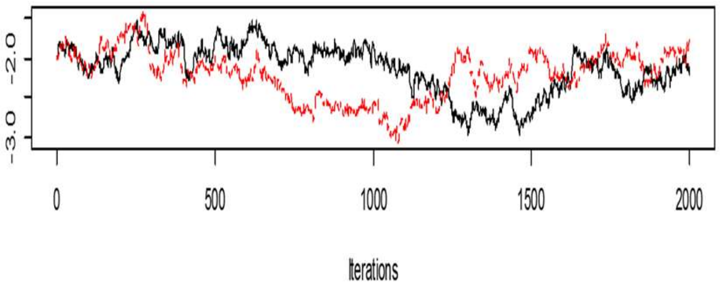
Turek et al. 2017
Monnahan et al. 2017
Monnahan et al. 2019

- Convergence - Scale Reduction factor (Gelman Rubin) R

- Efficient Sample Size - *ESS*
  $\approx$ Number of “independent” draws in the posterior sample

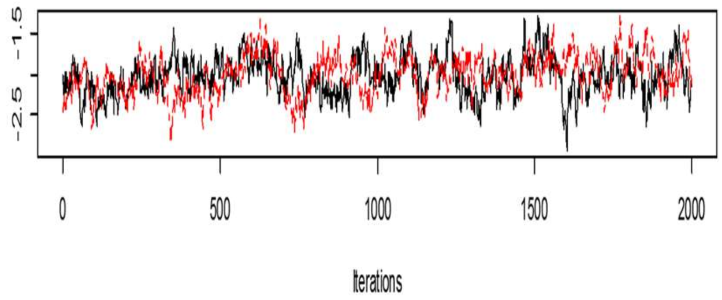
Poor mixing
High autocorrelation

ESS << n_{MCMC}



Good mixing
Low autocorrelation

ESS ~ n_{MCMC}



brary(coda), effective.size(), applied to post-burnin and post-thining sample

Computational efficiency

Turek et al. 2017
Monnahan et al. 2017
Monnahan et al. 2019

Trade-offs between algorithmic efficiency and run time

- $$\frac{Alg.\,efficiency}{run\ time} = \frac{ESS}{run\ time}$$

Exclude compilation time but include burnin

- $time_{ESS=1000}$ = Time required to obtain *ESS* = 1000

Comparing different configurations

NIMBLE

- Benchmarking
- Baseline version (with “good” inits)
- Model structure and parameterization
  - Deterministic transitions
  - Customized distributions to integrate out transitions
  - Prior for variance-covariance matrix
- Playing with block sampling

Choose initial values near the posterior

A "trick" that nobody wants to use in theory but that everyone uses in practice

➔ Easy to do in practice with Nimble

Simulate Nimble model with "good" parameters to produce appropriate inits of all latent states

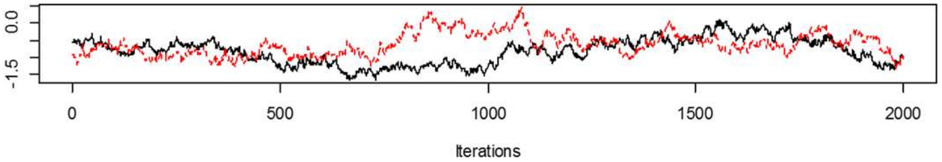
NIMBLE

```
mymod$theta_to_fix  <- value
mymod$simulate(nodes = Nodes_to_simulate)
```

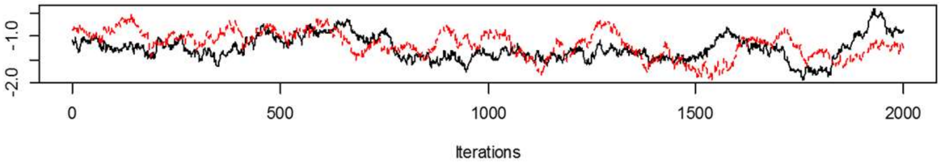
➔ Drastically reduces the number of MCMC draws to be discarded

Baseline

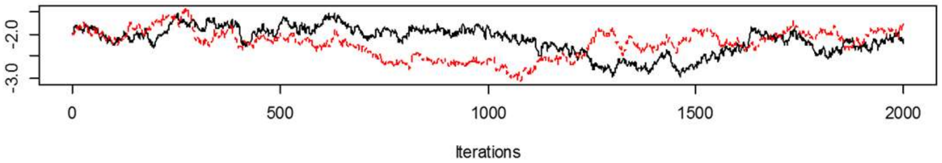
θ_3[t=10, s=10]



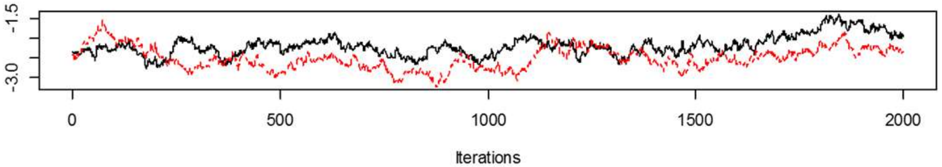
θ_3[t=20, s=10]



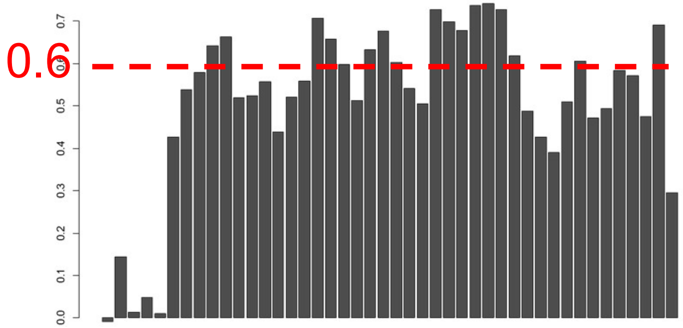
θ_3[t=30, s=10]



θ_3[t=40, s=10]

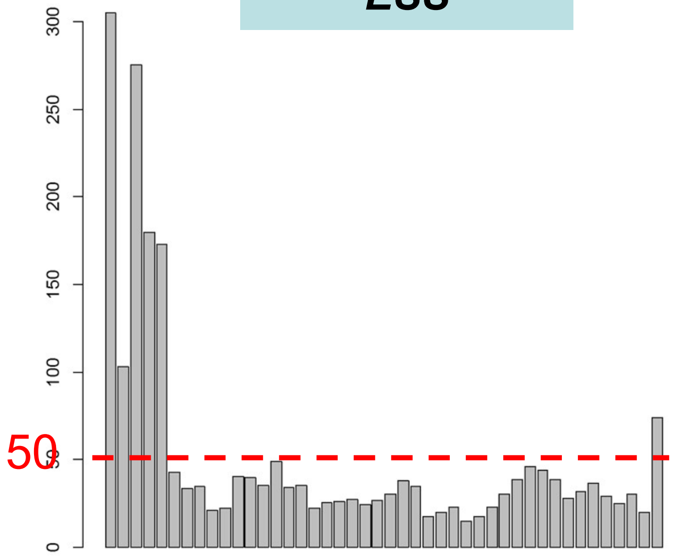


Autocor lag 1

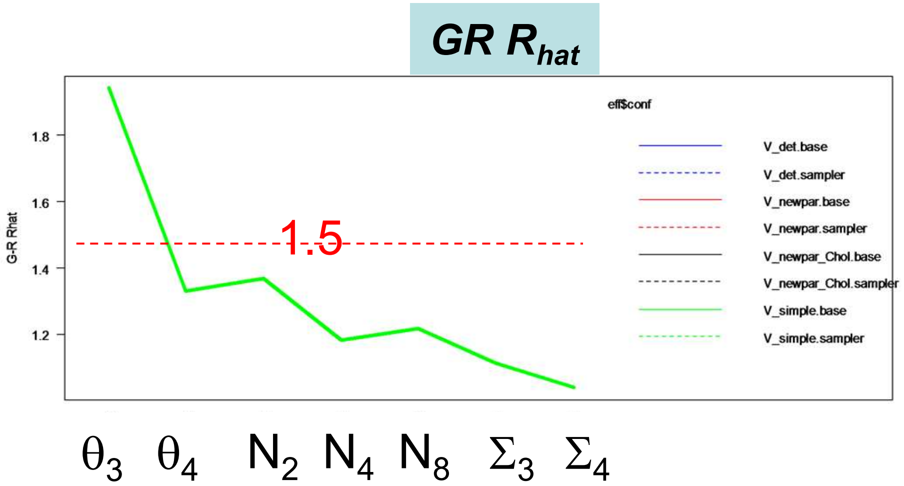


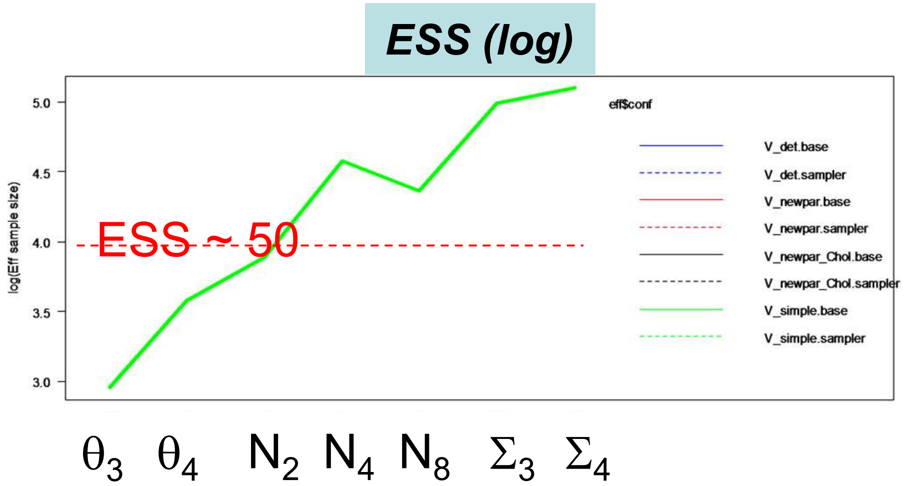
θ_3[t=1:45, s=10]

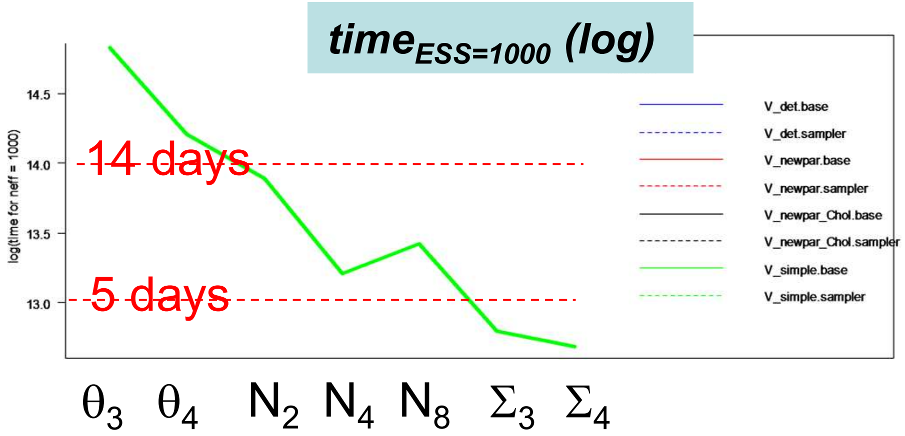
ESS



θ_3[t=1:45, s=10]







- ➔ Time to reach ESS=1000 is prohibitive
- ➔ MCMC behavior is heterogeneous among nodes

Comparing different configurations

NIMBLE

- Benchmarking
- Baseline version (with “good” inits)
- Model structure and parameterization
  - Deterministic transitions
  - Customized distributions to integrate out transitions
  - Prior for variance-covariance matrix
- Playing with block sampling

Core process equations

- **Stochasticity and synchronicity in**
  - **. Marine survival**
  - **. Proportion of fish maturing after one year spent at sea**

$$logit(\theta_{t+\mathbf{1},\mathbf{1}:\boldsymbol{S}}) \sim MultiNorm^{S}\left(logit(\theta_{t,\mathbf{1}:\boldsymbol{S}}), \Sigma\right)$$

Multi-variate random walk
Dimension S = 24

- **Demographic stochasticity**

$$\log(N_{i+1,t+1,s}) \sim N\left(\log(\theta_{i,t,s} \cdot N_{i,t,s}), \sigma\right)$$

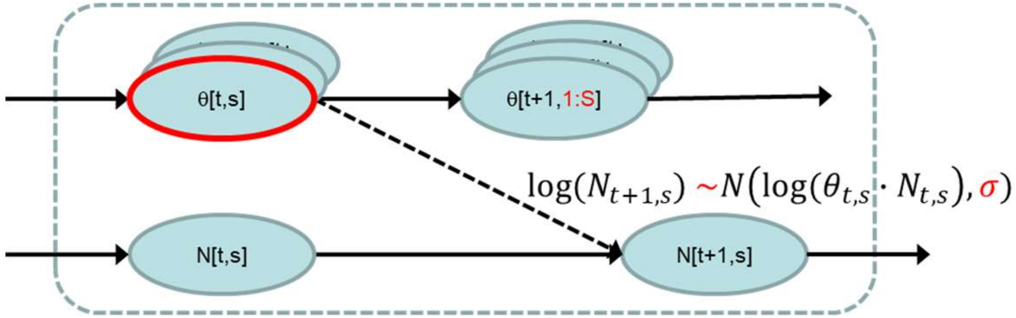
σ fixed to low value

Adding stochastic noise is a common recipe …

- Gibbs makes use of local dependency
  Updating $\theta_{t,s}$ → $\theta^*_{t,s}$ involves nodes directly connected to $\theta_{t,s}$

- Lognormal noise limits local dependencies → faster run

NIMBLE
```
> my.compileNimble$getDependencies(c("logit.theta[t,s]"))
```

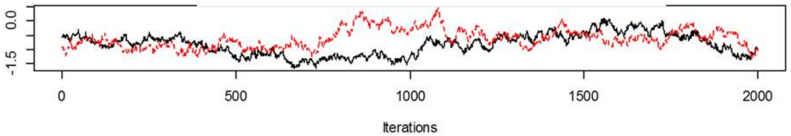


- … but penalizes algorithmic efficiency : Because $\boldsymbol{\sigma}$ is very low, only little change is authorized between $\boldsymbol{\theta_{t,s}}$ and $\boldsymbol{\theta^*_{t,s}}$ → high autocorrelation
  - $\theta^*_{t,s}$ = MCMC proposal for $\theta_{t,s}$
  - Its acceptance as a new MCMC sample depends on the ratio of the conditional probability
    $$[N_{t+1,s}|N_{t,s},\theta^*_{t,s},\sigma] / [N_{t+1,s}|N_{t,s},\theta_{t,s},\sigma]$$

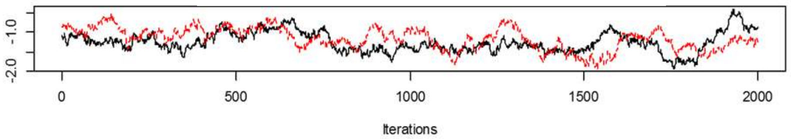
Effect of using deterministic transitions

Baseline

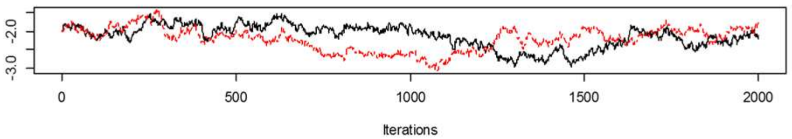
θ_3[t=10, s=10]



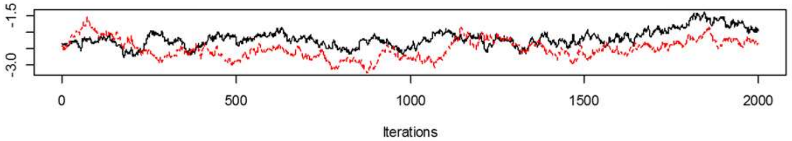
θ_3[t=20, s=10]



θ_3[t=30, s=10]



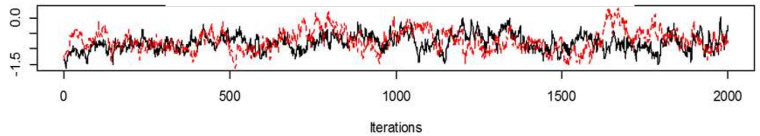
θ_3[t=40, s=10]



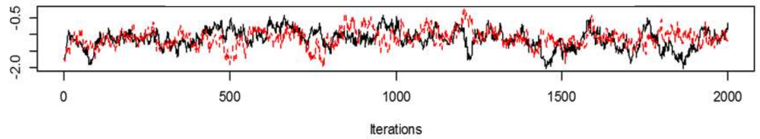
Deterministic transitions

➔ Improves mixing (algorithmic efficiency)

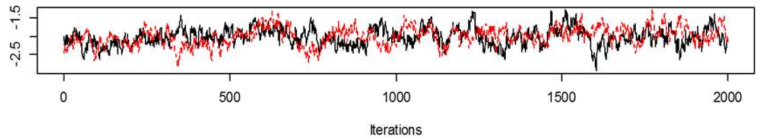
θ_3[t=10, s=10]



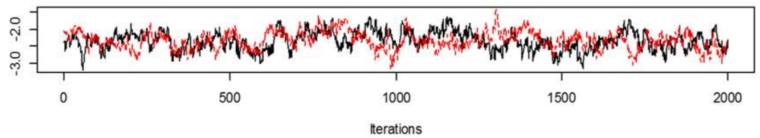
θ_3[t=20, s=10]



θ_3[t=30, s=10]



θ_3[t=40, s=10]



Effect of using deterministic transitions

➔ But dramatically increases computational requirements

NIMBLE

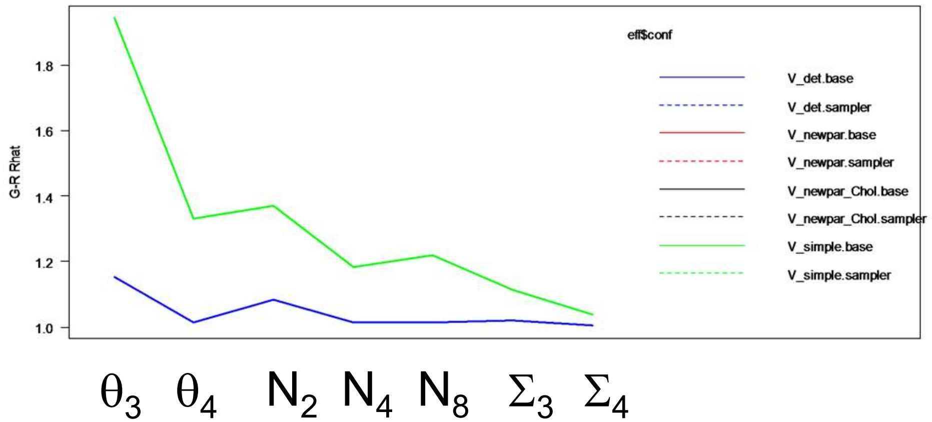
```
> my.compileNimble$getDependencies(c("logit.theta3[10,1]"))
```

```
 [1] "theta3[10, 1]"
 [2] "logit.theta3[11, 1:24]"
 [3] "N4[11, 1]"
 [4] "N5[11, 1]"
 [5] "N8[11, 1]"
 [6] "C5.NAC.1[11, 1]"
 [7] "C5.NAC.2[11, 1]"
 [8] "C5.NAC.3[11, 1]"
 [9] "N6[11, 1]"
[10] "C8.NAC.1[11, 1]"
[11] "N8.1[11, 1]"
[12] "C5.NAC.2.lab[11]"
[13] "N7[11, 1]"
[14] "Chw.1SW[11, 1]"
[15] "Chw.1SW.delSp[12, 1]"
[16] "lifted_log_oPN6_oBt_comma_r_cB_cP_L223[11, 1]"
[17] "C8.2[11, 1]"
[18] "N8.2[11, 1]"
[19] "C5.NAC.1.tot[11]"
[20] "C5.NAC.3.tot[11]"
[21] "C8.NAC.1.tot[11]"
[22] "lifted_log_oPC5_dot_NAC_dot_2_dot_lab_oBt_cB_cP_L247[11]"
[23] "N1[11, 1]"
[24] "lifted_log_oPChw_dot_1SW_oBt_comma_r_cB_cP_L229[11, 1]"
[25] "log.R1SW.m[11, 1]"
[26] "C8.NAC.3[12, 1]"
[27] "C8.NAC.4[12, 1]"
[28] "C8.NAC.5[12, 1]"
[29] "N9[12, 1]"
[30] "lifted_log_oPC5_dot_NAC_dot_1_dot_tot_oBt_cB_cP_L245[11]"
[31] "lifted_log_oPC5_dot_NAC_dot_3_dot_tot_oBt_cB_cP_L248[11]"
[32] "lifted_log_oPC8_dot_NAC_dot_1_dot_tot_oBt_cB_cP_L250[11]"
[33] "C8.2.tot[11]"
[34] "log.C1.tot.Lb.m[11]"
[35] "log.N2.m[11, 1]"
[36] "log.hwC1SW.m[11, 1]"
[37] "C8.NAC.4.lab[12]"
[38] "N10[12, 1]"
[39] "Chw.2SW[12, 1]"
[40] "Chw.2SW.delSp[13, 1]"
[41] "lifted_log_oPN9_oBt_comma_r_cB_cP_L226[12, 1]"
[42] "N7[12, 1]"
[43] "log.C1.Nf.3_7.m[11]"
[44] "log.C1.SPM.m[11]"
[45] "log.C1.nm.LbNf.m[11]"
[46] "C8.NAC.3.tot[12]"
[47] "C8.NAC.5.tot[12]"
[48] "mu.Gld[11, 1]"
[49] "mu.Gld[11, 2]"
[50] "mu.Gld[11, 3]"
[51] "mu.Gld[11, 4]"
[52] "mu.Gld[11, 5]"
[53] "mu.Gld[11, 6]"
[54] "mu.Gld[11, 7]"
[55] "mu.Gld[11, 8]"
[56] "mu.Gld[11, 9]"
[57] "mu.Gld[11, 10]"
[58] "mu.Gld[11, 11]"
[59] "mu.Gld[11, 12]"
[60] "mu.Gld[11, 13]"
[61] "mu.Gld[11, 14]"
[62] "mu.Gld[11, 15]"
[63] "mu.Gld[11, 16]"
[64] "mu.Gld[11, 17]"
[65] "mu.Gld[11, 18]"
[66] "mu.Gld[11, 19]"
[67] "mu.Gld[11, 20]"
[68] "mu.Gld[11, 21]"
[69] "mu.Gld[11, 22]"
[70] "mu.Gld[11, 23]"
[71] "mu.Gld[11, 24]"
[72] "lifted_log_oPC8_dot_2_dot_tot_oBt_cB_cP_L238[11]"
[73] "N2[11, 1]"
[74] "lifted_log_oPC8_dot_NAC_dot_4_dot_lab_oBt_cB_cP_L255[12]"
[75] "lifted_log_oPChw_dot_2SW_oBt_comma_r_cB_cP_L232[12, 1]"
[76] "log.R2SW.m[12, 1]"
[77] "N1[12, 1]"
[78] "lifted_log_oPC8_dot_NAC_dot_3_dot_tot_oBt_cB_cP_L252[12]"
[79] "lifted_log_oPC8_dot_NAC_dot_5_dot_tot_oBt_cB_cP_L257[12]"
[80] "prop_Gld[11, 1:24]"
[81] "log.CG2.m[11]"
[82] "Surv.eggs[11, 1]"
[83] "log.C2.tot.Lb.m[12]"
[84] "log.hwC2SW.m[12, 1]"
[85] "log.N2.m[12, 1]"
[86] "N10[13, 1]"
[87] "log.C2.Nf.3_7.m[12]"
[88] "log.C2.SPM.m[12]"
[89] "N2[12, 1]"
[90] "Surv.eggs[12, 1]"
[91] "N1[13, 1]"
[92] "log.N2.m[13, 1]"
[93] "N2[13, 1]"
[94] "Surv.eggs[13, 1]"
```

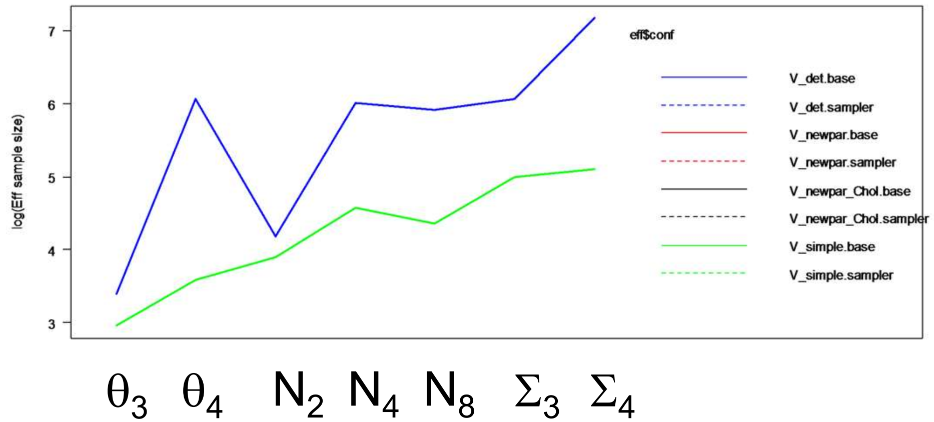
Effect of using deterministic transitions

Base (stochastic)
Determinitic

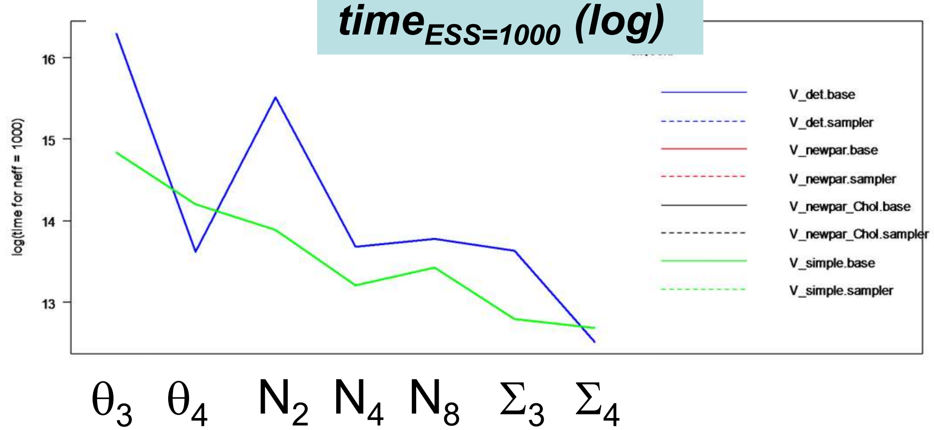
GR R_{hat}



ESS (log)



$time_{ESS=1000}$ (log)



- ➔ Improves algorithmic efficiency (mixing and ESS)
- ➔ Increases run time (~ x 7)
- ➔ ~ No gain in computational efficiency

Comparing different configurations

- Benchmarking
- Baseline version (with “good” inits)
- Model structure and parameterization
  - Deterministic transitions
  - Customized distributions to integrate out transitions
  - Prior for variance-covariance matrix
- Playing with block sampling

Customized distribution

- **Baseline (deterministic)**

Key transitions operate in 2 sequential steps

(1) $\theta_{t,1:s}$ is a multivariate random walk (logit scale)

$$logit(\theta_{t,1:S}) \sim MVNormal(logit(\theta_{t-1,1:S}), \Sigma)$$

(2) $N_{t+1,s} = \theta_{t,s} \times N_{t,s}$

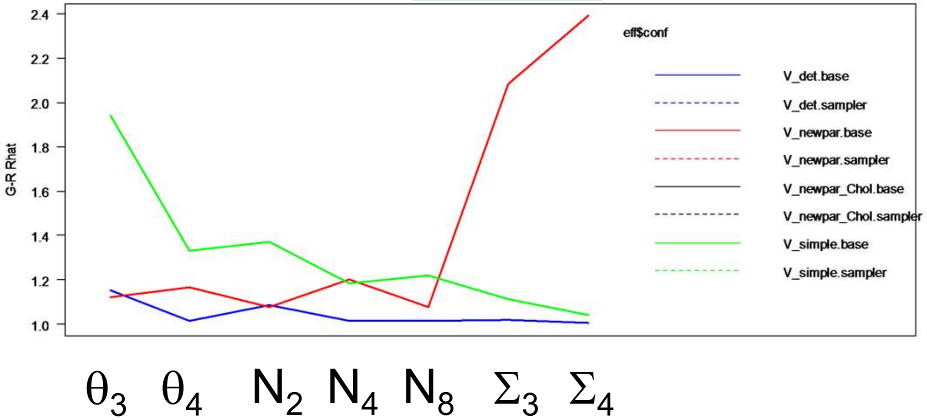
- **Customized distributions**

Build a customized sampling distribution that integrates two steps, to sample $log(N_{t+1,1:S})$ in its pdf :

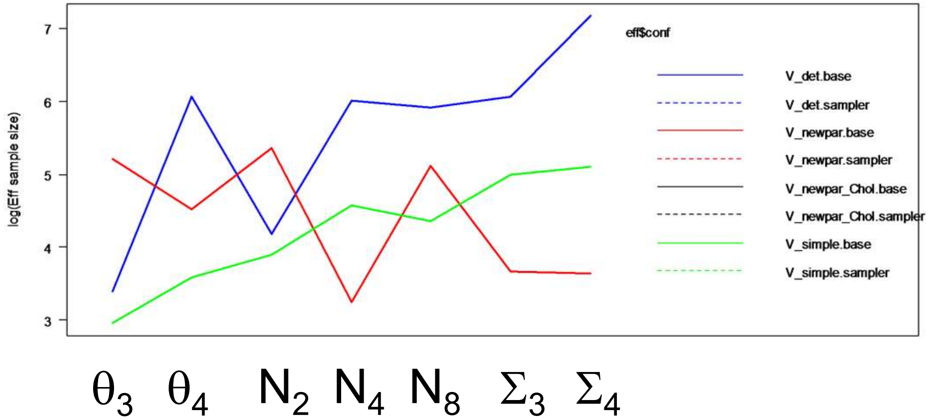
$$\sim \; log(N_{t+1,1:S}) \mid log(N_{t,1:S}), logit(\theta_{t-1,1:S}), \Sigma$$

Effect of using customized distributions

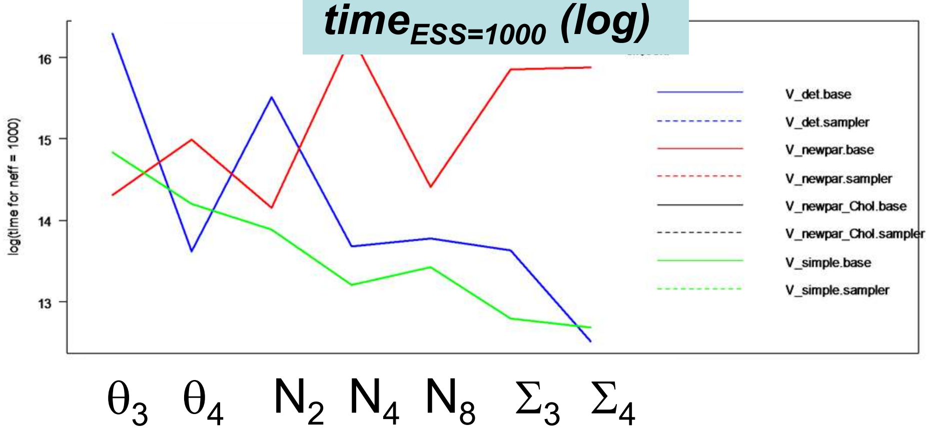
GR R_{hat}



ESS (log)



$time_{ESS=1000}$ (log)



- ➔ Run time ~ unchanged
- ➔ Improves mixing (algorithmic efficiency) for some but NOT all nodes (very low mixing for var-covar)

Comparing different configurations

NIMBLE

- Benchmarking
- Baseline version (with “good” inits)
- Model structure and parameterization
  - Deterministic transitions
  - Customized distributions to integrate out transitions
  - Prior for variance-covariance matrix
- Playing with block sampling

Block sampling

- **Block sampling is advocated as an efficient solution to improve MCMC efficiency**

Turek, D. et al. 2017. *Bayesian Analysis*, *12*(2), 465-490.
Ponisio et al., 2020. *Ecology and Evolution*, *10*(5), 2385-2416.

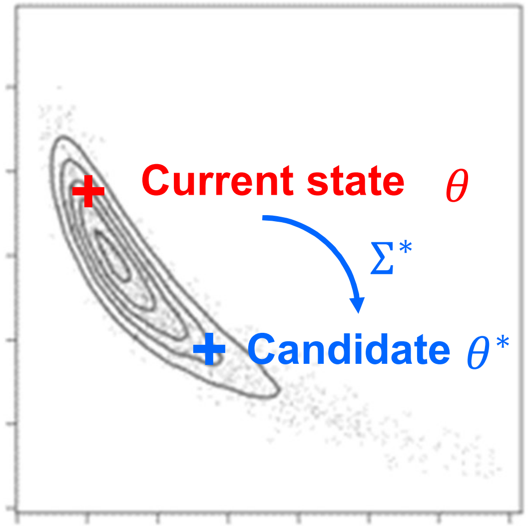
- **But efficiency of RW Metropolis Block sampling is a trade-off**

- **Gain in efficiency** to explore joint posterior distribution

  Propose candidates that accounts for the covariance of mult var. nodes

- **Loss of efficiency**, due to the difficulty to tune the var-covar matrix for the proposal (Σ^*)

  Scales with the dimension of block sampler (Turek et al. 2017)



Block sampling

- **By default, NIMBLE sets up an Adaptive RW Metropolis Block Sampler (dim = 24) for the multivariate nodes**

  $logit(\theta_{t,1:S})$ (*V_simple and V_det*)

  $logit(N_{t,1:S})$ (*V_newpar*)

- **Default blocking might be inefficient**

  → **Must be broken**

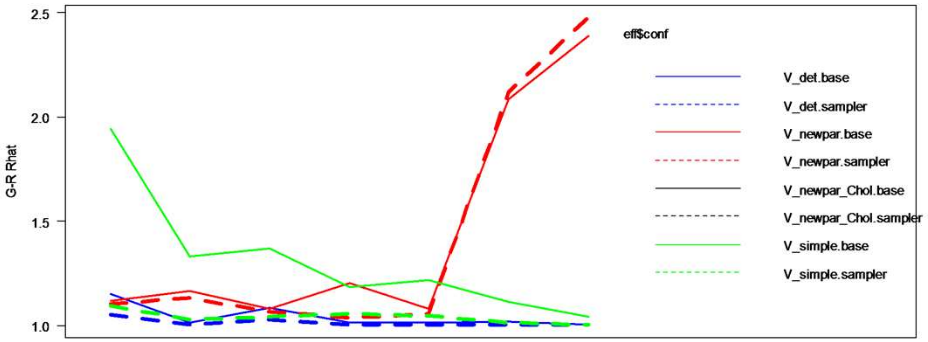
  → **Reconstruct customized blocks**

  Formulate strategies based on possible correlations between the parameters

Effect of forcing scalar ARW-sampler

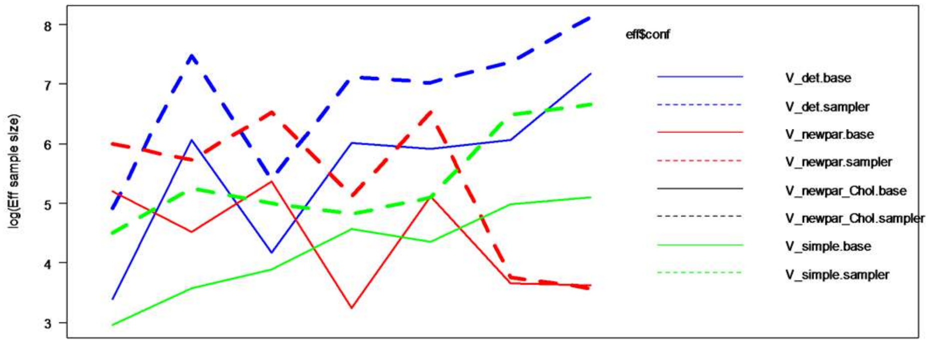
Block sampling
No block sampling

GR R_{hat}



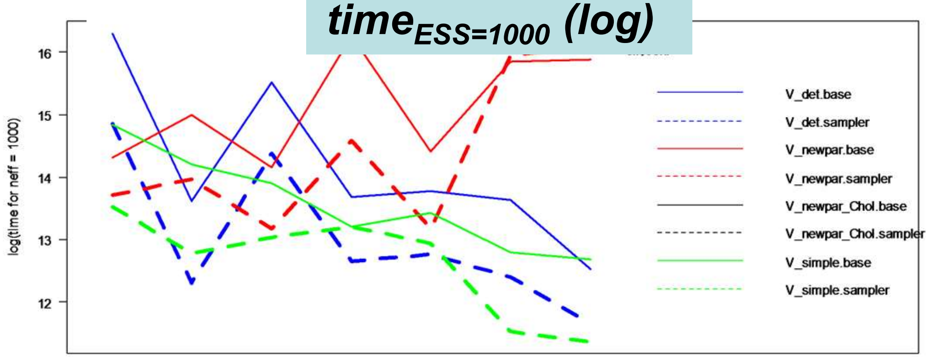
θ_3 θ_4 N_2 N_4 N_8 Σ_3 Σ_4

ESS (log)



θ_3 θ_4 N_2 N_4 N_8 Σ_3 Σ_4

$time_{ESS=1000}$ (log)



θ_3 θ_4 N_2 N_4 N_8 Σ_3 Σ_4

- ➔ Improves algorithmic efficiency
- ➔ Only slightly increases run time
- ➔ Improves computational efficiency (~ x 3) for all parameterizations

Take home messages

Conclusions

- NIMBLE revealed a flexible tool to explore strategies to improve MCMC

- Case study: Results are not (yet !) really concluding

  - Deterministic transitions
  - Customized distributions to integrate out transitions
  - Prior for variance-covariance matrix
  - Playing with block sampling

- Effect of different strategies depend on model nodes

Conclusions

- Identifying strategies to improve MCMC performance is becoming increasingly crucial as the complexity of models, and the run times to fit them, increases

Not an easy task …

- There is no one-size-fits-all best strategy, but rather problem-specific best strategies related to model structure and type
- Interactions between model formulation and sampling strategy on MCMC efficiency increase with model complexity
- Substantive improvement can be obtained through a cocktail of solutions

Thanks !